

AN EXPERT SYSTEM APPROACH TO THE OPTIMAL DESIGN OF
SINGLE-JUNCTION AND MULTIJUNCTION TANDEM SOLAR CELLS

By
CHUNE-SIN YEH

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1988

UNIVERSITY OF FLORIDA LIBRARIES

ACKNOWLEDGEMENTS

The author wants to express his deep gratitude to the chairman of his supervisory committee, Dr. Sheng S. Li, for his guidance and encouragement through the entire research process. The author also appreciates the other members of his supervisory committee, Drs. Arnost Neugroschel, Dorothea Burk, Gijs Bosman and Yuan-Chieh Chow for their participation on the committee.

Special thanks are given to Dr. R. Y. Loo for the irradiated solar cell samples and measurements and to Mrs. Li, his host family Mr. and Mrs. David Wilmot, and Mrs. Anne White for their kindness to his family. The author is also grateful to his colleagues at the Device Characterization and DLTS Lab for their helpful discussions. The financial support of the Universal Energy Systems Incorporation is greatly appreciated.

Finally, the author thanks his parents and family, especially his late father, for their love, expectation and patience throughout his graduate study.

TABLE OF CONTENTS

	Page
ACKNOWLEDGEMENTS	ii
ABSTRACT	vi
CHAPTERS	
ONE INTRODUCTION	1
1.1 Introduction	1
1.2 Fabrication Technology of Solar Cells	2
1.3 A Synopsis of this Research	3
TWO MODELING OF PROTON AND ELECTRON IRRADIATED SOLAR CELLS	6
2.1 Introduction	6
2.2 Displacement Defects	7
2.2.1 Defect Formation by Proton Bombardment	8
2.2.2 Defect Formation by Electron Bombardment	10
2.3 Degradation Calculation of Short-Circuit Current	13
2.4 Degradation Calculation of Open-Circuit Voltage	15
2.5 Degradation Calculation of Conversion efficiency	16
2.6 Results and Discussion	17
2.7 Summary	20
THREE A NEW METHOD FOR OPTIMAL DESIGN OF GAAS SINGLE-JUNCTION SOLAR CELLS	36
3.1 Introduction	36
3.2 Device Modeling for GaAs P/N Junction Solar Cells	37
3.3 Effects of Extrinsic Parameters on Device Modeling	40
3.3.1 Effect of Antireflection Coating	41
3.3.2 Effect of Grid Design	41
3.3.3 Effect of Series Resistance	42

3.3.4 Effect of High Sun Insolation	43
3.3.5 Effect of Irradiation	43
3.4 Constrained Optimization Technique	43
3.5 Optimal Design of GaAs Single-Junction Solar Cells	45
3.6 Summary	47
FOUR AN EXPERT SYSTEM APPROACH TO THE OPTIMAL DESIGN OF MULTIJUNCTION SOLAR CELLS	56
4.1 Introduction	56
4.2 Device Modeling of Multijunction Tandem Solar Cells	57
4.3 Concept of the Expert System Approach	59
4.3.1 Problem Formulation	60
4.3.2 Optimization and Heuristic Rules	63
4.4 Results and Discussion	64
4.5 Summary	67
FIVE THEORETICAL CALCULATIONS OF ELECTRON AND HOLE MOBILITIES	78
5.1 Introduction	78
5.2 Scattering Processes of $Al_xGa_{1-x}As$	79
5.2.1 Polar Optical Scattering	79
5.2.2 Piezoelectric Scattering	81
5.2.3. Deformation Potential Scattering	81
5.2.4. Ionized Impurity Scattering	82
5.2.5. Space Charge Scattering	82
5.2.6. Alloy Scattering	83
5.2.7. Intervalley Scattering	83
5.3. Analysis and Discussion	84
5.4. Summary	87
SIX SUMMARY, CONCLUSION and RECOMMENDATIONS	97
6.1. Summary and Conclusion	97
6.2. Recommendations	99
APPENDIX A A COMPUTER PROGRAM FOR CALCULATING THE TOTAL NUMBER OF DISPLACEMENT DEFECTS	101
APPENDIX B A COMPUTER PROGRAM FOR CALCULATING THE DEGRADATION OF SHORT-CIRCUIT CURRENT	111

APPENDIX C	AN EXPERT SYSTEM PROGRAM FOR OPTIMAL DESIGN OF SINGLE-JUNCTION AND MULTIJUNCTION TANDEM SOLAR CELLS	143
APPENDIX D	INPUT PARAMETERS FOR THE OPTIMAL DESIGN OF (ALGA)AS/GAAS/IN _{0.53} GA _{0.47} AS THREE-JUNCTION SOLAR CELLS	182
REFERENCES	188
BIOGRAPHICAL SKETCH	194

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

AN EXPERT SYSTEM APPROACH TO THE OPTIMAL DESIGN OF
SINGLE-JUNCTION AND MULTI-JUNCTION TANDEM SOLAR CELLS

By

Chune-Sin Yeh

August 1988

Chairman: Sheng-San Li

Major Department: Electrical Engineering

The use of an expert system approach to the optimal design of single-junction and multijunction solar cells is a potential new design tool in photovoltaics. This dissertation presents the results of a comprehensive study of this new design method. To facilitate the realistic optimal design of the two-terminal monolithic single-junction and multijunction tandem solar cells, a rule-based system was established by adopting the experimental data and/or semiempirical formulae used today for those design parameters. A numerical simulation based on the displacement damage theory was carried out to study the degradation of AlGaAs/GaAs solar cells after proton or electron irradiation. The damage constant of the minority carrier diffusion length, an important design parameter of a solar cell for space application, was calculated. An efficient Box complex optimization technique with minor modifications is analyzed and applied to accelerate the convergence rate of the algorithm. Design rules were implemented in order to reduce the search space of the optimal design and to make a compromise in the tradeoff between the conflicting criteria for selection. The computation time for the optimal design is very much reduced by adopting these

rules. Realistic optimal design of single-junction and multijunction solar cells were obtained and verified from the expert system and then compared with the state of the art technology. Finally, theoretical calculations of the electron and hole mobilities of $\text{Al}_x\text{Ga}_{1-x}\text{As}$ were performed. The results show that electron mobilities of $\text{Al}_x\text{Ga}_{1-x}\text{As}$ are quite sensitive to the Al composition, temperature, doping density and defect density. This knowledge is indispensable to the fabrication of a high efficiency AlGaAs top cell of a GaAs based multijunction solar cells.

CHAPTER 1 INTRODUCTION

1.1 Introduction

For the past three decades, researchers around the world have been involved in studying the degradation of solar cells induced by energetic electrons or protons for space applications. As the number of satellite launches has increased dramatically, the need for a space power system using solar cells has increased significantly. Moreover, by knowing the degradation of performance in the electron or proton irradiated solar cells, one is able to design a solar cell that gives an optimum end of life conversion efficiency for the specific space mission. This should lead to innovations in design and fabrication such as shallow junctions, window layers and multijunction solar cells.

The objective of this dissertation is to use an expert system approach to study the optimum design of single-junction and multijunction tandem solar cells for both space and terrestrial applications. It should be noted that because the theoretical conversion efficiency increases very slowly beyond three-junction tandem cells [1] and the technology of fabricating cells beyond three junctions is difficult in practice [2], the multijunction study in this work was focused only on the two-terminal monolithic structures of single-junction, two-junction and three-junction solar cells.

Although recently multijunction solar cells have become increasingly important for space applications, the multijunction cell structure in itself is not a new concept. In 1955, Jackson [3] proposed that the efficiency of solar cells could be increased significantly by stacking one or more cells composed of different semiconductor materials. Since then, many other researchers [4-26] have proposed varying approaches to the optimal design of the multijunction cells, theoretically and/or experimentally.

However, little progress in multijunction solar cell design has been made due to (1) lack of reliable data on the electrical and optical parameters of the solar cell materials, (2) lack of adequate device modeling, (3) inefficient optimization algorithm. In an effort to provide an optimal and realistic design of multijunction solar cells for space and terrestrial applications, the research reported here makes use of an expert system approach.

In this chapter an overview of the major solar cell fabrication technologies now in use is presented. Different processes may have varying effects on the quality and efficiency of solar cells. Finally, a synopsis of our research is given.

1.2 Fabrication Technology of Solar Cells

Although the potential performance advantages of the GaAs single-junction and III-V compounds based multijunction solar cells have been discussed [2-22], their practical realization remains a problem, due to the absence of a good fabrication technology. Among those fabrication processes developed for III-V compound, liquid phase epitaxy (LPE), molecular beam epitaxy (MBE) and metal organic chemical vapor deposition (MOCVD) have proven successful [19, 27-31] and hence the use of the multijunction structure in solar cells has become important and promising.

However, since there is no single fabrication technique which is superior to the other two in terms of crystal purity, doping level, throughput and controllability of cell thickness and uniformity, it is still too early to say which of these three technologies will dominate the future development of single-junction and/or multijunction solar cells. Therefore, in this brief review, it seems most useful to describe the advantages and weaknesses of each fabrication process, but not to give details about the processes themselves.

LPE is the reference technology for the growth of epitaxial layer in III-V compounds. It offers high purity crystal and excellent transport properties, minimum contamination, uniform thickness control and reductions in point defect and disloca-

tion. It remains an easy and cheap method for growing high efficiency AlGaAs-GaAs solar cells. However, the throughput is small.

MOCVD is a standard system used for making high efficiency III-V solar cells. It is a flexible technology that allows the growth of a quite complicated multilayer structures with different doping levels and different compositions. It can grow very thin layers with precise doping and thickness. And MOCVD has large throughput. However, materials made by MOCVD may have carbon contamination and high defect density.

The advantages of MBE are similar to those of MOCVD. While, MBE has only a medium throughput, it can produce high purity crystal without contamination. In multijunction solar cells, the most important feature of MOCVD and MBE is the ease of incorporating superlattice or tunnel junction into the cascade solar cell structures. This can reduce the dislocation density in the top cell and hence reduce the recombination loss and increase output voltage.

1.3 A Synopsis of this Research

Based on the survey given in this chapter, it would appear that the realistic optimal design of single-junction and multijunction solar cells could be realized through the establishment of the expert system. Moreover, it should be apparent that the lack of full knowledge of the material parameters and their relations — mobility, diffusion length, etc. — as well as the complexity of the optimization technique, would limit the development of a general analytical model which accounts for all variations of the design parameters. It is the goal of this research to generate an expert system for solving this optimal design problem.

The first step was to set up a rule-based knowledge data base for the design parameters listed above by adopting the experimental data and/or semiempirical formulae in use today. This reduced the complexity of the numerical simulation required to solve the problem. The next step was the development of a model of the

proton or electron irradiated solar cells. The model developed here is based on the displacement damage model. A computer program for calculating the degradations of short-circuit current, open-circuit voltage and conversion efficiency of a solar cell was coded. This program included the calculations of the damage constant of the minority carrier diffusion length. This parameter is important for the optimal designs of irradiated solar cells.

In addition, an efficient optimization algorithm was implemented for the expert system. Some heuristic rules were applied to reduce the search space of the design problem and to make some compromise in the tradeoff between the conflicting criteria of selection. Realistic optimal designs were obtained from the simulations and their performance compared with the state of the art technology.

In Chapter 2, a numerical model of proton and electron irradiated solar cells is presented. In this model it is assumed that the radiation induced displacement defects form an effective recombination center which reduces the minority carrier diffusion length and hence degrades the short-circuit current I_{sc} , open-circuit voltage V_{oc} and conversion efficiency η_c of the solar cell. It should be noted that although the numerical model applies to a special case which is limited to the normally incident protons or electrons, this model can be extended to simulate the real space environment where the incoming protons or electrons are omnidirectional.

In Chapter 3, a new computer model for truly optimizing the structure of GaAs single-junction solar cell for both space and terrestrial application is proposed. The model, however, can apply easily to other solar cell systems. It not only takes into account the effects of the intrinsic structural parameters such as junction depth X_j , cell thickness T_j , doping densities N_A and N_D , surface recombination velocities S_p and S_n , but also incorporates the extrinsic structure parameters. And an efficient Box optimization algorithm [32] with minor modifications is implemented.

In Chapter 4, an expert system approach to the optimal design of multijunction

solar cells for both terrestrial and space applications is described for the first time. An expert system is a knowledge-intensive computer program. The knowledge of an expert system consists of facts and rules. The facts constitute a body of information that is widely shared, publicly available and generally agreed upon by experts in the field [33-34]. The rules are those if-then rules that characterize expert level decision making in the field. In general, a good and robust expert system must include as many facts and rules as possible. However, because of the availability of the earlier research reports and the tradeoff between the number of rules and the computation time, this expert system is currently limited to the AlGaAs, GaAs, $\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$, Si and Ge materials.

In Chapter 5, the theoretical calculations of the electron and hole mobilities in $\text{Al}_x\text{Ga}_{1-x}\text{As}$ are reported. It was found that the mobilities of $\text{Al}_x\text{Ga}_{1-x}\text{As}$ are quite sensitive to the Al composition, temperature, doping density and defect density. This knowledge is indispensable to the fabrication of a high quality and high efficiency AlGaAs top cell of the GaAs based multijunction solar cells. The final chapter, Chapter 6, summarizes the materials and offers suggestions for future research.

In addition to the six chapters in this dissertation, there are four appendices. In Appendix A, a computer program for calculating the total number of displacement defects induced by the energetic proton and electron bombardments is presented. A computer program for manipulating the degradation of short-circuit current, open-circuit voltage, conversion efficiency and damage constant of the minority carrier diffusion length is in Appendix B. An expert system program for optimal design of single-junction and multijunction solar cells is in Appendix C. Finally, the input parameters for (AlGa)As/GaAs/(InGa)As three-junction solar cells are shown in Appendix D.

CHAPTER 2

MODELLING OF PROTON AND ELECTRON IRRADIATED SOLAR CELLS

2.1 Introduction

The advantages offered by single junction GaAs solar cells for space applications are their high power conversion efficiency, their radiation hardness and their relative insensitivity to temperature compared to silicon solar cells. Although GaAs solar cells have not been extensively flown in space, radiation experiments performed on earth have demonstrated their superior hardness as compared to silicon cells. In this chapter we model the radiation degradation characteristics of GaAs single-junction solar cells and GaAs based multijunction cells under proton or electron radiation environments.

An improved numerical model is offered for computing the displacement defect density, the damage constants of the minority carrier diffusion lengths and the degradations of the short-circuit current I_{sc} , open-circuit voltage V_{oc} and the conversion efficiency η_c in proton or electron irradiated single-junction and multijunction solar cells is presented. The model which we use in this study is based on the displacement damage theory in semiconductors. It is assumed that the radiation induced displacement defects form effective recombination centers which reduce the minority carrier diffusion length and hence degrade the I_{sc} , V_{oc} and η_c of the solar cell. It should be noted that although our numerical model applies to a special case, limited to the normally incident protons or electrons, this model can be extended to simulate the real space environment where the incoming protons or electrons are omnidirectional.

In earlier work, Wilson et al. [35] and Young [36] have also used the displacement damage model to study the GaAs solar cells. We have extended their work

and have obtained a better correlation between theory and experimental findings by including both electron and hole capture cross sections in our calculations. The electron and hole capture cross sections were determined for the GaAs and AlGaAs cells after proton or electron irradiation using deep level transient spectroscopy (DLTS) measurements. In addition, the model includes calculations of the normalized open-circuit voltage, conversion efficiency and damage constant of the minority carrier diffusion length. The damage constant of the minority carrier diffusion length for the irradiated solar cell is one of the important parameters in optimal design of a solar cell for space application. In addition to the single-junction GaAs solar cell, a simple model was developed for calculating the displacement damage in the proton and electron irradiated $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}/\text{GaAs}$ two-junction solar cell and in the $\text{Al}_{0.35}\text{Ga}_{0.65}\text{As}/\text{GaAs}/\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$ or Ge three-junction solar cell.

The calculations indicate that the degradation rate in each cell varies greatly and depends critically not only on the energy, fluence and the direction of the incident proton or electron, but also on the thickness of each cell in the multijunction cells. Excellent agreement was obtained between our calculated values and the measured I_{sc} , V_{oc} and η_c in proton or electron irradiated AlGaAs and GaAs solar cells for proton energies from 100 KeV to 10 MeV and fluences from 10^{10} to 10^{12} cm^{-2} , and for 1 MeV electron and fluences varying from 10^{14} to 10^{16} cm^{-2} under normal incidence conditions.

2.2 Displacement Defects

A solid may be affected in two ways by the energetic particle bombardment [37]. First, the lattice atoms may be removed from their regular lattice sites and produce displacement damage. Second, the irradiating particles may cause change in the chemical properties of the solid via ion implantation or transmutation. In our model, it is assumed that the dominant defect produced by the incident electrons or protons is due to lattice displacement. Under this assumption, an atom will invariably

be displaced from its lattice site during collisions if its kinetic energy exceeds the threshold energy, T_d for the atomic displacement, and will not be displaced if its kinetic energy is less than T_d [38].

2.2.1 Defect Formation by Proton Bombardment

When energetic protons collide with atoms, the energy transferred to the struck atoms is the most important consideration in evaluating irradiation damage [38]. The number of defects formed by an energetic proton coming to rest in the solar cell are related to the energy of the proton, the transferred energy and the threshold energy, T_d of the solar cell. Given the conservation of energy and momentum, it follows that the maximum energy which can be transferred to the struck atom in a primary head-on collision with energy, E , is

$$T_M = \frac{4M_1M_2E}{(M_1 + M_2)^2} \quad (2.1)$$

where E is the initial energy of the proton and M_1 and M_2 are the masses of proton and struck atom respectively. This transferred energy may range from zero in a glancing collision to a maximum T_M in a head-on collision. As for a proton, the energy transferred in a collision can be calculated by ignoring the screening effect. Therefore, the scattering in a proton collision obeys the Rutherford differential cross section $d\sigma_p$ which is given by

$$d\sigma_p = C \frac{dT}{T^2} \quad (2.2)$$

$$C = 4\pi a_0^2 \frac{M_1}{M_2} Z_1^2 Z_2^2 \frac{E_r^2}{E} \quad (2.3)$$

where T is the transferred energy. M_1 and M_2 are the same as those in equation 2.1, Z_1 and Z_2 are the atomic number of the proton and struck atom respectively, E_r is

the Rydberg constant and a_0 is the Bohr radius. Since the defects occur when the energy transferred is greater than T_d , the displacement cross section σ_p is given by

$$\sigma_p = \int_{T_d}^{T_M} d\sigma_p = C \left(\frac{1}{T_d} - \frac{1}{T_M} \right) \quad (2.4)$$

The average energy transferred, \bar{T} , in a Rutherford collision which displaces atoms can be calculated as follows:

$$\begin{aligned} \bar{T} &= \frac{\int_{T_d}^{T_M} T d\sigma_p}{\int_{T_d}^{T_M} d\sigma_p} \\ &= \frac{T_d T_M}{T_M - T_d} \ln \left(\frac{T_M}{T_d} \right) \end{aligned} \quad (2.5)$$

If the transferred energy is sufficiently large ($T \gg T_d$), additional displacement can be produced by the recoiling nucleus before it comes to rest at an interstitial site. The average recoil displacements, v , produced by one initiating proton collision event is given as a function of T_M , on the assumption that half of the recoil energy produces further displacement and the other half is dissipated in other processes described in [35, 38]

$$V_p(E) = \begin{cases} 1.0 & \text{for } T_d < T_M < 2T_d \\ 1 + \frac{T_M}{2(T_M - T_d)} \ln \left(\frac{T_M}{T_d} \right) & \text{for } 2T_d \leq T_M \end{cases} \quad (2.6)$$

Since the mass of the proton is heavier than that of an electron, the velocity of an energetic proton is slower than an electron with the same energy. Thus, a proton has the potential of multiple scattering before coming to rest. The displacement defects $D(E_0)$ induced by an energetic particle under normal incident condition are

$$DP(E_0) = \int_0^P N \sigma_P V_P dP \quad (2.7)$$

where P is the path length traveled by a proton in coming to rest, and all the other variables have the same definitions as in Eqns. 2.4 and 2.6. As for the omnidirectional incident irradiation, the total displacement defects D_{om} are given by [39]

$$DP_{om} = 2\pi \int_0^{E_0} dE \int_0^1 d(\cos \theta) [DP(E_0) - DP(E_0(x/\cos \theta))] \quad (2.8)$$

where θ is the incident angle of the particle toward the solar cell.

2.2.2 Defect Formation by Electron Bombardment

Because of the small mass of an electron, the electron must travel at a relatively high velocity in order to produce displacement. The maximum energy which can be transferred in a collision by an electron with mass m and kinetic energy E is

$$T_M = \frac{2(E + 2mC^2)E}{M_2C^2} \quad (2.9)$$

where C is the velocity of light, and m and M_2 are masses of the electron and struck atom respectively. Consequently, the nonrelativistic Rutherford scattering is inadequate for the electron. Relativistic Coulomb scattering has been treated by McKinley-Feshbach as follows [40]:

$$d\sigma_e = \frac{4\pi(a_0Z_2E_r)^2(1-\beta^2)}{m^2C^4\beta^4} \left[1 - \beta^2 \frac{T}{T_M} + \pi Z_2\beta \left(\sqrt{\frac{T}{T_M}} - \frac{T}{T_M} \right) / 137 \right] T_M \frac{dT}{T^2} \quad (2.10)$$

where β is the electron velocity ratio to the velocity of light C . All other variables have the same definition as those in the previous equations. Integration of equation 2.10 yields the displacement cross section for an incident electron:

$$\sigma_e = \frac{4\pi(a_0 Z_2 E_r)^2 (1 - \beta^2)}{m^2 C^4 \beta^4} \left[\frac{T_M}{T_d} - 1 - \beta^2 \ln(T/T_M) + 2\pi a \beta \left(\sqrt{\frac{T_M}{T_d}} - 1 \right) - \pi a \beta \ln(T_M/T_d) \right] \quad (2.11)$$

where a is equal to $Z_2/137$. The average energy transferred during a collision is

$$\begin{aligned} \bar{T} &= \frac{\int_{T_d}^{T_M} T d\sigma_e}{\int_{T_d}^{T_M} d\sigma_e} \\ &= \frac{T_M \ln(T_M/T_d) - \beta^2(T_M - T_d) + 2\pi a \beta (T_M - \sqrt{T_M - T_d}) - \pi a \beta (T_M - T_d)}{T_M/T_d - 1 - \beta^2 \ln(T/T_M) + 2\pi a \beta \left(\sqrt{\frac{T_M}{T_d}} - 1 \right) - \pi a \beta \ln(T_M/T_d)} \end{aligned} \quad (2.12)$$

Thus, if the transferred energy is sufficiently large ($T \gg T_d$), the mechanism of recoil displacement produced by an electron is similar to that of a proton. The average recoil displacements, V_e , produced by one initiating electron collision event are given [35, 38]

$$V_e(E) = \begin{cases} 0.0 & \text{for } \bar{T}(E) < T_d \\ 1 & \text{for } T_d \leq \bar{T} \leq 2T_d \\ 1 + \frac{\bar{T}(E)}{2T_d} & \text{for } 2T_d < \bar{T}(E) \end{cases} \quad (2.13)$$

The displacement defects $D(E_0)$ induced by an energetic electron under normal incident condition are

$$DE(E_0) = \int_0^R N \sigma_e V_e dR \quad (2.14)$$

where R is the penetration depth of an electron in coming to rest. Since the electron

mass is small, we can neglect the effects of multiple scattering. As for omnidirectional incident irradiation, the total displacement defects D_{om} are given by [39]

$$DE_{om} = 2\pi \int_0^{E_0} dE \int_0^1 d(\cos \theta) [DE(E_0) - DE(E_0(x/\cos \theta))] \quad (2.15)$$

Now, according to Eqns. 2.7, 2.8, 2.14 and 2.15, for numerical calculations of the total number of displacement defects, the threshold energy, path length, penetration depth (range) and the reduced energy of protons or electrons after travelling an x distance must be given. Although the threshold energies for GaAs and Ge are given as 9.5 eV and 27.5 eV respectively, those of InGaAs and AlGaAs are still unknown. In this respect, a linear extrapolation has been made to calculate the two unknown values as follows:

$$T_d(Al_xGa_{1-x}As) = 0.5 \times [T_d(Al) \times x + T_d(Ga) \times (1 - x) + T_d(As)] \quad (2.16)$$

and

$$T_d(In_xGa_{1-x}As) = T_d(InAs) \times x + T_d(GaAs) \times (1 - x) \quad (2.17)$$

In addition to the threshold energy, the path length or range an electron or proton coming to rest is also unknown. Thus, we adopt the path length and the range from the data given recently by Janni [41] for the proton case and by Page et al. [42] for the electron case. Since these figures are only given for elements, and the GaAs, (AlGa)As and (InGa)As are compound materials, approximations were made in calculating path lengths and ranges based on the following assumptions [41]:

$$\frac{1}{P_c} = \sum \frac{W_i}{P_i} \quad (2.18)$$

$$\frac{1}{R_c} = \sum \frac{W_i}{R_i} \quad (2.19)$$

where P_c and R_c are the path length and range of the compound materials respectively, R_i and P_i are data for each element and W_i is the weighting function of each element. The least square method was employed to fit those data in order to obtain the expressions for P and R for the solar cells studied here.

As for the reduced energy, E_{re} , of protons or electrons after penetrating a distance x , the fitting process is similar to those for path length and range. These empirical formulae were obtained through the computer programs described in Appendices A and B.

2.3 Degradation Calculation of Short-Circuit Current

To derive an expression for the short-circuit current in an irradiated cell, the following simplified assumptions were made [35, 43, 44]: (1) radiation-induced defects do not greatly alter internal cell electric field, (2) radiation-induced defects alter the cell operation mainly through change in minority carrier lifetimes in the bulk, and (3) radiation-induced displacements within the solar cell form recombination centers for minority carriers of electron-hole pairs produced by photon absorption.

When sunlight impinges on a solar cell, the short-circuit current generated from a solar cell is given by

$$I_{sc}(\lambda) = \int_0^t \eta(x) \rho(x, \lambda) dx \quad (2.20)$$

where $\eta(x)$ is the current collection efficiency, t is the cell thickness, and

$$\rho(x) = K\alpha \exp(-\alpha x) \quad (2.21)$$

is the photon generation rate at depth x , K is the integrated solar photon flux in the absorption band and α is the absorption coefficient.

After proton or electron irradiations, defects are created within the materials and the solar cell's short-circuit current is decreased. To rewrite the expression for the short-circuit current after proton or electron irradiation, I'_{sc} , we need to add a loss term $[1 - F(x)]$ to the integral of Eqn. 2.20, to account for the carrier recombinations caused by the newly generated defects. The expression for the recombination loss coefficients, $F(x)$, is [35]

$$F(x) = 1 - E_2[\sqrt{6}\sigma_r\phi|D(E_x) - D(E_{x_j})|] \quad (2.22)$$

and

$$I'_{sc}(\lambda) = \int_0^t \eta(x)(1 - F(x))\rho(x, \lambda)dx \quad (2.23)$$

where $E_2(z)$ is the exponential integral of order 2, σ_r is the electron or hole capture cross section, ϕ is the proton or electron fluence, E_x is the reduced proton or electron energy after penetrating a distance, x and x_j is the junction depth.

The normalized I_{sc} degradation can thus be calculated by using the expression

$$\frac{I'_{sc}}{I_{sco}} = \int_{\lambda_1}^{\lambda_2} \frac{I_{sc}(\lambda)d\lambda}{I_{sco}(\lambda)d\lambda} \quad (2.24)$$

λ_1 and λ_2 denote the shortwave and long-wave limits of the total useful solar spectra for the solar cell.

2.4 Degradation Calculation of Open-Circuit Voltage

Once the damage constants of the minority carrier lifetimes or diffusion lengths are known, calculations of V_{oc} degradation are straightforward. According to the Shockley, Read and Hall theory [45], the minority carrier lifetime is inversely proportional to the defect density, N_t and is given by [45]

$$\tau_{n,p} = \frac{1}{N_t V_{th} \sigma_{n,p}} \quad (2.25)$$

where V_{th} is the thermal velocity and $\sigma_{n,p}$ is the capture cross-section. Thus, the minority carrier lifetime of the solar cell after proton or electron irradiation, $\tau_{p,n}'$, can be calculated from Eqn. 2.25 by substituting the values of the displacement damage density N_t and the capture cross-sections from the DLTS measurements. It should be noted that N_t can be calculated directly from Eqns. 2.7, 2.8, 2.14 and 2.15 for proton and electron irradiations respectively. Knowing $\tau_{p,n}'$, the damage constants of the minority carrier lifetimes K_{τ_p} and K_{τ_n} are given by [46, 47] as

$$\frac{1}{\tau_p'} = \frac{1}{\tau_p} + K_{\tau_p} \phi \quad (2.26)$$

$$\frac{1}{\tau_n'} = \frac{1}{\tau_n} + K_{\tau_n} \phi \quad (2.27)$$

Now using the relation that $L^2 = D \tau$, it is easy to obtain the damage constants of the minority carrier diffusion lengths K_{L_n} and K_{L_p} . The open-circuit voltage before irradiation is expressed by

$$V_{oc} = \frac{nK_B T}{q} \ln\left(\frac{I_{sc}}{I_0} + 1\right) \quad (2.28)$$

where n is the diode ideality factor and I_0 is the saturation current of the diode. K_B is

the Boltzman constant and q is the electron charge. In general, the current conduction mechanism through the diode is dominated by diffusion or by recombination currents. When the diffusion current dominates, the diode's ideality factor, n , is equal to 1, and I_0 is given by

$$I_0 = (qn_i^2 A) \left(\frac{D_p}{L_{po} N_D} + \frac{D_n}{L_{no} N_A} \right) \quad (2.29)$$

where n_i is the intrinsic carrier density, A is the area of a solar cell, D_p and D_n are the hole and electron diffusion coefficients, respectively, L_{po} is the hole diffusion length and L_{no} is the electron diffusion length and N_D and N_A are the donor and acceptor dopant densities. When the recombination current at the junction dominates, the ideality factor, n , is equal to 2 and the saturation current, I_0 , is

$$I_0 = \frac{qn_i W A}{\sqrt{\tau_p \tau_n}} \quad (2.30)$$

where W is the depletion width, and τ_p and τ_n are the hole and electron lifetimes in the n-region and p-region, respectively. In general, the n value will vary between 1 and 2 because both mechanisms contribute to the transport mechanism.

As for the open-circuit voltage of the irradiated cell, V'_{oc} , it is given by

$$V'_{oc} = \frac{nK_B T}{q} \ln \left(\frac{I'_{sc}}{I'_0} + 1 \right) \quad (2.31)$$

where I'_0 is the saturation current after irradiation.

2.5 Degradation Calculation Of Conversion Efficiency

The maximum power conversion efficiency of a solar cell is the product of the short-circuit current, open-circuit voltage and the fill factor. Experimentally it has

been observed that the fill factor of the cell remains unchanged after irradiation [48]. Thus, the normalized conversion efficiency after proton or electron irradiation is

$$\frac{\eta_c'}{\eta_{co}} = \frac{I'_{sc} V'_{oc}}{I_{sco} V_{oco}} \quad (2.32)$$

2.6 Results and Discussion

Figure 2.1 shows the baseline design of the GaAs solar cell. The GaAs and $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}$ solar cells were fabricated at Hughes Research Laboratories (HRL) by using infinite solution LPE technique. The base region of the solar cell is n-type ($1 \times 10^{17} \text{ cm}^{-3}$). A wide bandgap, p^+ , ($2 \times 10^{18} \text{ cm}^{-3}$), Be doped AlGaAs window layer was grown on the top of the base layer to passivate the GaAs or $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}$ surface and to reduce its surface recombination velocity. During the growth of the window, Be was diffused into the base region to form an electrical junction. The thickness of the window layer and the junction depth were measured to be 0.35 and 0.5 μm , respectively. The GaAs and $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}$ solar cells' conversion efficiency, before the irradiation, were tested at AM0, 1 sun condition to be 16.0% to 17% and 7.5% to 8.0 % respectively. The proton irradiation experiments for various proton energies and fluences were performed at HRL (low energy protons), California Institute of Technology (medium energy protons) and University of California, Davis (high energy protons). The details of the experiments and results were given in earlier publications [49-53].

Table 2.1 lists all the cell parameters used in calculating the degradation of the short-circuit current and these parameters are identical to the actual GaAs and $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}$ solar cells used in the proton and electron irradiation experiments. Figures 2.2 and 2.3 give the calculated proton range and the number of displacement damage defects per incident proton as a function of proton energy. For example, a 300 KeV proton will penetrate 2 μm deep into GaAs solar cell (Fig. 2.2) and produces about 100 defects per cm^3 (Fig. 2.3). Figure 2.4 shows the remaining short-circuit

current after proton irradiation as a function of the proton energy. The solid lines are the calculated short-circuit currents and the circles are the experimental values. Examining both Fig. 2.2 and Fig. 2.4, it is clear that only proton energies greater than 50 KeV are seen to create damage to the cell. The 100 KeV protons, which are stopped at about $0.8 \mu\text{m}$ below the surface, create damage close to the junction. The 200 KeV protons, which are stopped at about $1.35 \mu\text{m}$ below the surface cause damage throughout the p-region and the junction. The 290 KeV protons, which are stopped at about $2.0 \mu\text{m}$ below the surface produce damage in the bulk of the n-GaAs layer. Protons with energies greater than 1 MeV will pass through the entire cell, hence, will create less damage to the cell than the low energy protons. Thus, the calculated short-circuit current agrees closely with the experimental values.

The damage constants of the minority carrier diffusion lengths, K_{Ln} and K_{Lp} , were also deduced from these calculations and are given in Table 2.1. The open-circuit voltage after proton irradiations can then be calculated as a function of proton energies and fluences. Figure 2.5 gives the normalized open-circuit voltage, V_{oc}/V_{oc0} , as a function of the proton energy. Again, as with the results obtained for the short-circuit current, the low energy protons near 200 KeV degrade the open-circuit voltage more severely than other energies. This is because the 200 KeV protons are stopped in the vicinity of the pn junction. From Fig. 2.5, it can be seen that the theory correlates well with the experimental data. It shows that the modeling technique is valid for the GaAs case.

Table 2.3 summarizes the calculated and measured cell characteristics after proton irradiations. It is again gratifying to see that the theory and experimental data are in accord.

Figure 2.6 gives the total number of displacement defects as a function of incident electron energies for GaAs and $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}$ solar cells. It shows that the total number of defects induced by the electrons is much less than that induced by the

protons (Fig. 2.3) with the same energy. This is because the energy left to the cell from the incident electron is much less than that from the proton, owing to the small effective mass of the electron. Therefore, more electron fluences are needed to cause degradation of solar cells. Table 2.4 indicates the short-circuit degradation of 1-MeV electron irradiated GaAs and $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}$ p-n junction cells for three different fluences. It shows that the radiation hardness of $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}$ is greater than that of GaAs. Table 2.5 summarizes the calculated and measured cell degradation after 1-MeV electron irradiation for both normal incidence and omnidirectional incidence cases. The calculated results again show the strong agreements with the experimental data for electrons fluences of 10^{14} and 10^{15} cm^{-2} .

Figure 2.7 gives the flow chart for simulating the degradation of multijunction solar cells. In general, a middle cell or bottom cell may be affected by the incident energetic protons or electrons only when the reduced energy, after penetrating the top cell or middle cell respectively, is greater than zero. Therefore, in addition to the fluences and initial energy of the incident particle, the thickness of each cell is critical in calculating degradation of the multijunction solar cells. Since there are no experimental data on proton or electron irradiated multijunction solar cells available for comparisons, for simplicity the nearly optimal design of the $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}/\text{GaAs}$ two-junction solar cell was chosen for discussion in this section. The input parameters of these two-junction solar cells are same as those in Table 2.1. The short-circuit current degradation of proton irradiated $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}/\text{GaAs}$ two-junction cells are given in Table 2.6. Table 2.7 lists the I_{sc} degradation of 1-MeV electron irradiated $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}/\text{GaAs}$ two-junction solar cells.

Figure 2.8 shows the short-circuit current degradation of the proton irradiated $\text{Al}_{0.35}\text{Ga}_{0.65}\text{As}/\text{GaAs}/\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$ (or Ge) three-junction solar cell. The detailed discussion of these multijunction systems has been described in previous publications [54].

2.7 Summary

In this chapter a numerical model for computing the displacement damage for single-junction and multijunction solar cells has been developed and applied to the proton and electron irradiated single-junction GaAs and $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}$ solar cells, two-junction $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}/\text{GaAs}$ and $\text{Al}_{0.35}\text{Ga}_{0.65}\text{As}/\text{GaAs}/\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$ (or Ge) three-junction solar cells under different fluences and energies. Excellent agreement was obtained between our calculated values and the measured I_{sc} , V_{oc} and η_c in proton or electron irradiated (AlGa)As and GaAs solar cells for proton energies from 100 KeV to 10 MeV and fluences from 10^{10} to 10^{12} cm^{-2} , and for 1 MeV electrons and fluences varying from 10^{14} to 10^{16} cm^{-2} under normal incidence conditions. Moreover, it is shown in this chapter that in order to obtain an optimal multijunction solar cell with specified end of life efficiency, various physical parameters for each cell must be determined. It was pointed out that major difficulties encountered in carrying out the theoretical calculations using the model developed here include some unknown input parameters and the lack of experimental data to facilitate comparisons with the simulations. These uncertainties can be removed once the actual cell structures for the multijunction cells are fabricated and characterized.

Table 2.1 Input parameters for the simulations of proton or electron irradiated $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}$ and GaAs solar cells.

Electron or hole lifetime $\tau_{n,p}$ (s)	2×10^{-8}	4×10^{-9}
Electron diffusion length, L_n (μm)	6.0	3.41
Hole diffusion length, L_p	3.0	0.5
P-dopant density, N_A (cm^{-3})	2×10^{18}	2×10^{18}
N-dopant density, N_D	1×10^{17}	1×10^{17}
Intrinsic density, n_i	1.8×10^6	1.66×10^2
Electron capture cross section, σ_n (cm^2) (proton irradiated cell)	1.8×10^{-14}	5.0×10^{-16}
Hole capture cross section, σ_p	1.5×10^{-13}	5.0×10^{-15}
Electron capture cross section, σ_n (cm^2) (electron irradiated cell)	1.2×10^{-14}	2.0×10^{-15}
Hole capture cross section, σ_p	1.4×10^{-13}	2.0×10^{-14}
$\text{Al}_{0.85}\text{Ga}_{0.15}\text{As}$ window thickness, d (μm)	0.34	0.30
Junction depth, X_j (μm)	0.5	0.55
Cell thickness, t (μm)	10	4
Cell area, A (cm^2)	4	4
Threshold energy, T_d (eV)	9.5	10.5

Table 2.2 Calculated damage constants of the minority-carrier diffusion length in GaAs p-n junction solar cell.

Energy (MeV)	K_{L_n}	K_{L_p}
0.1	0.105	0
0.3	0.00081	0.34
1.0	0.00025	0.00835
2.0	0.00006	0.00273
5.0	0.000029	0.00169
10	0.000006	0.00109

Table 2.3 Calculated and experimental data of the degradation of I_{sc} , V_{oc} and η_c in proton irradiated (AlGa)As-GaAs solar cell.

Energy (MeV)	Fluence cm^{-2}	I_{sc}/I_{sc0} (%)	V_{oc}/V_{oc0} (%)	η_c/η_{c0} (%)
		Cal. Exp.	Cal. Exp.	Cal. Exp.
0.1	10^{10}	0.97 0.97	0.97 0.925	0.94 0.89
	10^{11}	0.80 0.81	0.72 0.81	0.61 0.63
	10^{12}	0.49 0.50	0.63 0.66	0.30 0.28
0.3	10^{10}	0.92 0.87	0.93 0.94	0.85 0.81
	10^{11}	0.74 0.71	0.89 0.86	0.67 0.62
	10^{12}	0.44 0.46	0.85 0.78	0.37 0.31
1.0	10^{10}	0.98 -	0.96 -	0.94 -
	10^{11}	0.95 -	0.92 -	0.88 -
	10^{12}	0.80 -	0.89 -	0.71 -
2.0	10^{10}	0.99 0.98	0.98 0.979	0.97 0.95
	10^{11}	0.96 0.938	0.96 0.94	0.93 0.90
	10^{12}	0.83 0.81	0.93 0.87	0.78 0.71
5.0	10^{10}	0.99 1.00	0.99 1.00	0.98 1.00
	10^{11}	0.97 0.93	0.96 0.97	0.93 0.90
	10^{12}	0.86 0.84	0.93 0.90	0.80 0.76
10.0	10^{10}	0.99 1.00	0.995 0.99	0.98 0.99
	10^{11}	0.97 0.96	0.975 0.97	0.945 0.95
	10^{12}	0.89 0.89	0.945 0.93	0.843 0.84

Table 2.4 I_{sc} degradation of one-MeV electron irradiated $Al_{0.33}Ga_{0.67}As$ and GaAs p-n junction solar cells.

Fluence	10^{16} cm^{-2}		10^{15} cm^{-2}		10^{14} cm^{-2}	
	Cal.	Exp.	Cal.	Exp.	Cal.	Exp.
normal incidence						
$Al_{0.33}Ga_{0.67}As$	0.695	-	0.925	0.926	0.986	0.986
GaAs	0.640	-	0.886	0.82	0.976	0.99
omnidirection						
$Al_{0.33}Ga_{0.67}As$	-	-	0.729	-	0.942	-
GaAs	-	-	0.245	-	0.660	-

Table 2.5 Calculated and experimental data of degradation of I_{sc} , V_{oc} , η_c and K_{Lp} and K_{Ln} in one-MeV electron irradiated $Al_{0.33}Ga_{0.67}As$ solar cell.

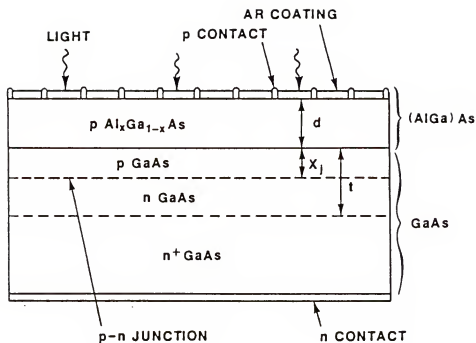
Fluence	10^{15} cm^{-2}		10^{14} cm^{-2}	
	Cal.	EXP.	Cal.	Exp.
normal Incidence				
I_{sc}/I_{sco}	0.925	0.926	0.986	0.986
V_{oc}/V_{oco}	0.934	0.97	0.991	0.989
η_c/η_{co}	0.863	0.848	0.977	0.941
K_{Ln}	9.9×10^{-8}	–	2.34×10^{-8}	–
K_{Lp}	4.3×10^{-6}	–	7.04×10^{-7}	–
Omnidirectional				
I_{sc}/I_{sco}	0.729	–	0.942	–
V_{oc}/V_{oco}	0.909	–	0.922	–
η_c/η_{co}	0.662	–	0.868	–
K_{Ln}	2.89×10^{-7}	–	2.12×10^{-7}	–
K_{Lp}	4.45×10^{-5}	–	4.08×10^{-5}	–

Table 2.6 I_{sc} degradation of proton irradiated $Al_{0.33}Ga_{0.67}As/GaAs$ two-junction solar cells.

E(MeV)	10^{10} cm^{-2}		10^{11} cm^{-2}		10^{12} cm^{-2}	
	Top	Bottom	Top	Bottom	Top	Bottom
0.1	0.993	1.00	0.969	1.00	0.829	1.00
0.3	0.992	1.00	0.963	1.00	0.832	1.00
1.0	0.996	0.992	0.994	0.965	0.980	0.843
2.0	0.996	0.993	0.995	0.974	0.984	0.871
5.0	0.996	0.994	0.995	0.981	0.987	0.900
10.0	0.996	0.995	0.996	0.988	0.992	0.941

Table 2.7 I_{sc} degradation of one-MeV electron irradiated $Al_{0.33}Ga_{0.67}As/GaAs$ two-junction solar cells.

Fluence	Top cell	Bottom cell
10^{16} cm^{-2}	0.780	0.64
10^{15} cm^{-2}	0.950	0.886
10^{14} cm^{-2}	0.990	0.976



NUMBER OF FINGERS = 24

p CONTACT: Au-Zn-Ag

n CONTACT: Au-Ge-Ni-Ag

AR COATING : Ta₂O_x

p Al_xGa_{1-x}As: $x \geq 0.85$

CELL SIZE = 2 x 2 cm²

Fig. 2.1 The cross sectional view of an (AlGa)As-GaAs p-n junction solar cell.

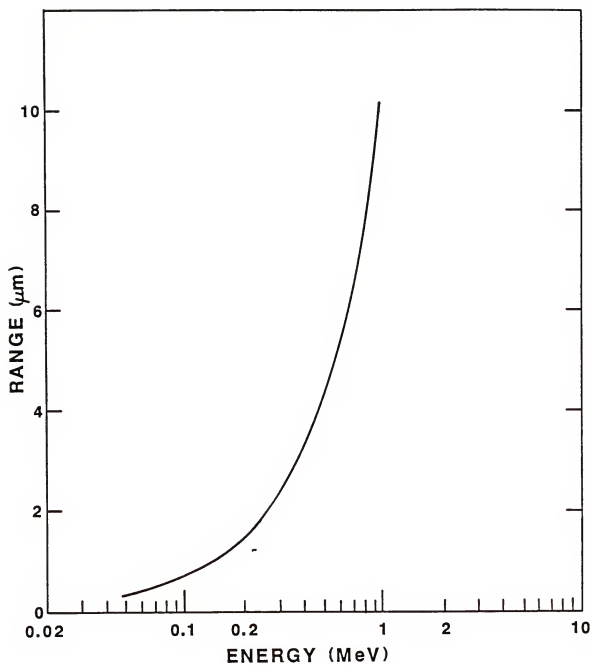


Fig. 2.2 The range of an $\text{Al}_{0.85}\text{Ga}_{0.15}\text{As-GaAs}$ solar cell vs. incident proton energies.

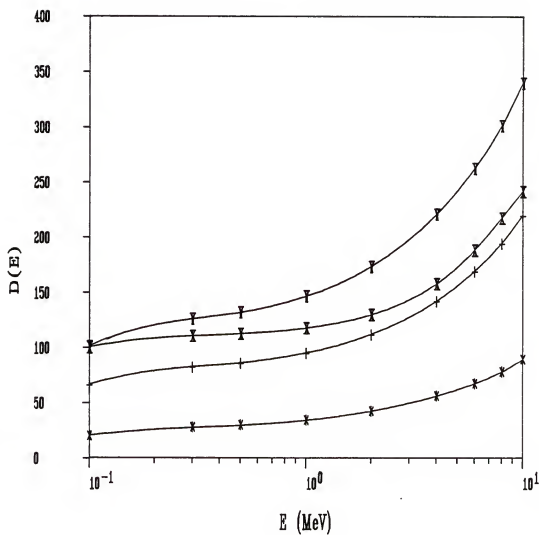


Fig. 2.3 The total number of displacement defects vs. incident proton energies for GaAs, Ge, (InGa)As and (AlGa)As single-junction solar cells. Y : $\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$, X : GaAs, + : $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}$ and * : Ge.

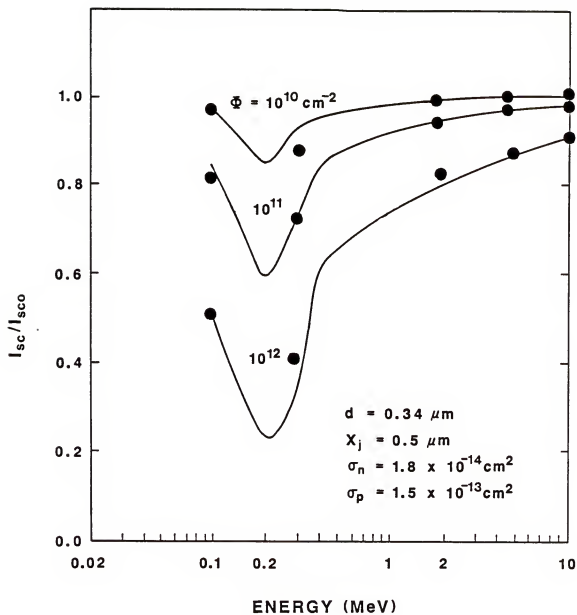


Fig. 2.4 The normalized short-circuit currents in the $\text{Al}_{0.85}\text{Ga}_{0.15}\text{As}$ -GaAs p-n junction solar cells. Solid curves are from our calculations; solid dots are the experimental data.

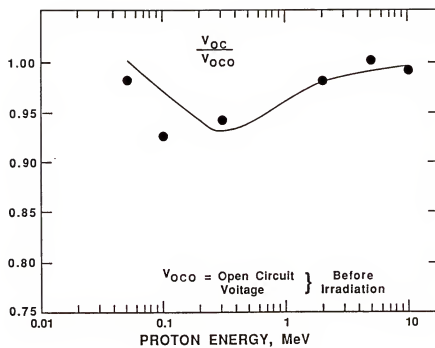


Fig. 2.5 The normalized open-circuit voltage in the $\text{Al}_{0.85}\text{Ga}_{0.15}\text{As}$ -GaAs p-n junction solar cell.

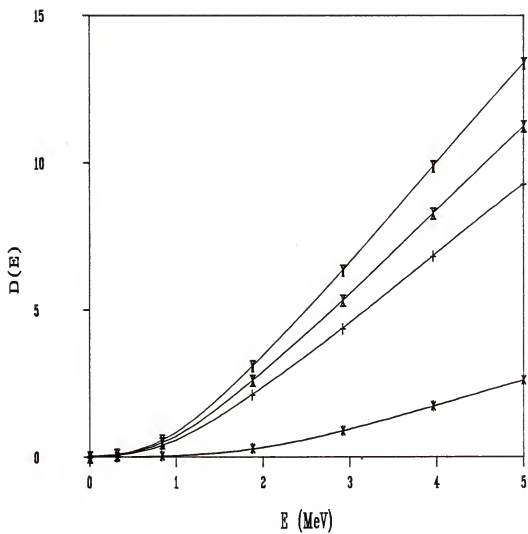


Fig. 2.6 The total number of displacement defects vs. incident electron energies for GaAs, Ge, (InGa)As and (AlGa)As single-junction solar cells. Y : $\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$, X : GaAs, + : $\text{Al}_{0.33}\text{Ga}_{0.67}\text{As}$ and * : Ge.

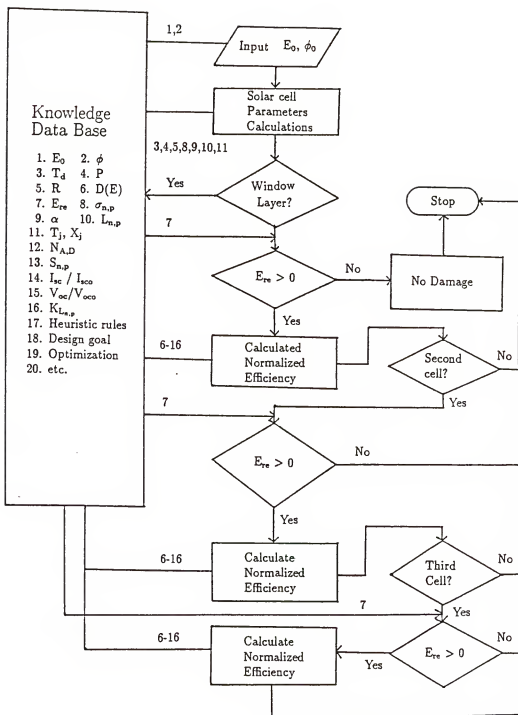


Fig. 2.7 The flowchart for simulating degradation of the irradiated multijunction solar cells.

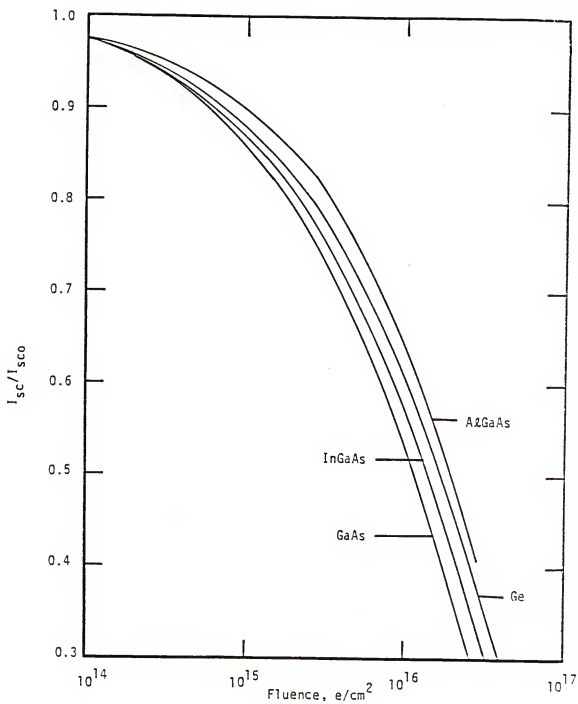


Fig. 2.8 I_{sc} degradation of the electron irradiated $\text{Al}_{0.35}\text{Ga}_{0.65}\text{As}/\text{GaAs}/\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$ (or Ge) multijunction solar cells.

CHAPTER 3 A NEW METHOD FOR OPTIMAL DESIGN OF GAAS SINGLE-JUNCTION SOLAR CELLS

3.1 Introduction

Since simple analytical models can not provide enough information and accuracy to optimize the design of solar cells [55], computer modeling has become an important and essential tool for solar cell design and analysis. The incorporation of an efficient optimum algorithm for computer simulation would enable, using the knowledge of cell parameters from the state of the art and its applications, to optimize the cell parameters in order to attain a maximum conversion efficiency.

Recently, Chen and Wu [56] and Fossum [57] have proposed two different computer aided designs for silicon solar cells while Hovel [20], Hamaker [21] and Kinechtli et al. [19] have established simplified solutions for GaAs solar cells. However, the results obtained by these previous models were less than truly optimum due both to ignorance of the interaction between the parameters and to the lack of an efficient optimum algorithm in computer simulation. In addition, they did not include the extrinsic parameters such as series resistance, contact structures, irradiation, air mass ratio, temperature, etc., in the optimization cycle and hence the results were not realistic.

In this chapter a new computer model is proposed for truly optimizing the structure of a GaAs single-junction solar cell for space application. However, the model can apply easily to other solar cell systems as well. Our model not only takes into account the effects of the intrinsic structural parameters such as junction depth X_j , cell thickness T_j , doping densities N_A and N_D , surface recombination velocity S_p and S_n , but also incorporates the extrinsic structural parameters. It should be noted that

the electron or proton irradiation on a solar cell for space application will degrade the cell's performance. Therefore, the damage constants of cell parameters, such as minority carrier diffusion length and minority carrier lifetime, should be estimated in advance in order to attain an accurate optimal structure of the cell. The detailed discussion of the calculations of damage constants of the irradiated solar cell has been presented in Chapter 2. In this chapter it is assumed that the values of damage constants for the cell parameters are available and hence can be employed directly in the optimum simulation.

To test the optimum algorithm both GaAs and Si solar cells have been simulated. It turns out that the conversion efficiency of a GaAs cell is better than that of Si. The results also show that there can be several optimum sets of parameters that yield nearly the same values for efficiency while maintaining acceptable characteristics.

Section 3.2 of this chapter outlines the device modeling for a p/n junction solar cell. The effects of extrinsic parameters on solar cell parameters are presented in section 3.3. In section 3.4 we propose a true and efficient optimization technique for the optimal design of a single-junction solar cell. Comparisons between the calculated results and the published data have been done also in this section. Summary is made in section 3.5.

3.2 Device Modeling for GaAs P/N Junction Solar Cells

For simplicity the discussion will be restricted to a p/n junction solar cell with uniform doping. The cross section diagram of the GaAs p/n junction solar cell to be analyzed is the same as that shown in Fig. 2.1. There are four major solar cell parameters involved namely short-circuit current I_{sc} , open-circuit voltage V_{oc} , conversion efficiency η_c and fill factor F.F. in designing a typical p/n junction solar cell. In general, the higher the values of the I_{sc} , V_{oc} and F.F. a solar cell can produce, the higher η_c the solar cell can obtain. However, a solar cell with a low energy bandgap tends to produce higher I_{sc} and lower V_{oc} than the cell with a higher energy bandgap.

This tradeoff complicates the design of a solar cell.

Theoretically, when the photon of the light impinges on the p/n junction solar cell having a bandgap energy, E_g , the short-circuit current of the solar cell can be represented as the sum of the currents obtained from the emitter layer I_E , base layer I_B and depletion region I_W as shown in Fig. 3.1. It is given as [20]

$$I_{sc} = \int_{0.35}^{\lambda_1} I_E(\lambda) + I_B(\lambda) + I_W(\lambda) d\lambda \quad (3.1)$$

where λ_1 (μm) = $1.24 / E_g$.

The current component from emitter layer is represented as

$$I_E(\lambda) = \frac{qA\rho(\lambda)(1-R(\lambda))\alpha L_n}{\alpha^2 L_n^2 - 1} (-\alpha L_n \exp(-\alpha X_j) + \frac{\alpha L_n + S_n \frac{\tau_n}{L_n} (1 - \exp(-\alpha X_j) \cosh \frac{X_j}{L_n}) - \exp(-\alpha X_j) \sinh \frac{X_j}{L_n}}{S_n \frac{\tau_n}{L_n} \sinh \frac{X_j}{L_n} + \cosh \frac{X_j}{L_n}}) \quad (3.2)$$

where A is area of the cell, q is the electron charge, $R(\lambda)$ is reflection coefficient, α is the absorption coefficient of the cell and is a function of wavelength, L_n is the electron diffusion length, τ_n is the electron lifetime and $\rho(\lambda)$ is the solar photon flux density which is [58]

$$\rho(\lambda) = 4.06 \times 10^{18} \lambda^3 H_\lambda \quad (3.3)$$

where H_λ is the solar radiance. The solar radiance is crucial in evaluating the conversion efficiency of a solar cell for different air mass ratio. Existing software, LOW-TRAN6 [59], is used here.

The current component from the base layer is

$$I_B(\lambda) = \frac{qA\rho(\lambda)(1-R(\lambda))\alpha L_p}{\alpha^2 L_p^2 - 1} \exp(-\alpha(X_j + W_j))$$

$$\left(\alpha L_p - \frac{S_p \frac{\tau_p}{L_p} \left(\cosh \frac{d'}{L_p} - \exp(-\alpha d') \right) + \sinh \frac{d'}{L_p} + \alpha L_p \exp(-\alpha d')}{S_p \frac{\tau_p}{L_p} \sinh \frac{d'}{L_p} + \cosh \frac{d'}{L_p}} \right) \quad (3.4)$$

where L_p is the hole diffusion length, τ_p is the hole lifetime, W_j is the depletion width and $d' = T_j - X_j - W_j$.

The current component from the depletion region is shown as

$$I_W = qA\rho(\lambda)(1-R(\lambda)) \exp(-\alpha X_j) (1 - \exp(-\alpha W_j)) \quad (3.5)$$

The open-circuit voltage V_{oc} for the solar cell is given by

$$V_{oc} = \frac{nK_B T}{q} \ln\left(\frac{I_{sc}}{I_0} + 1\right) \quad (3.6)$$

where K_B is the Boltzmann constant, T is the absolute temperature and I_0 is the saturation current density, n is the diode ideality factor. For simplicity, we may assume that I_0 is either a simple diffusion-dominated current if n is close to 1 or a recombination-dominated current if n is close to 2. The former one is given as

$$I_0 = qAn_i^2 \frac{D_p}{L_p N_D} \left(\frac{S_p \cosh(d'/L_p) + \frac{D_p}{L_p} \sinh(d'/L_p)}{\frac{D_p}{L_p} \cosh(d'/L_p) + S_p \sinh(d'/L_p)} \right) +$$

$$qAn_i^2 \frac{D_n}{L_n N_A} \left(\frac{S_n \cosh(X_j/L_n) + \frac{D_n}{L_n} \sinh(X_j/L_n)}{\frac{D_n}{L_n} \cosh(X_j/L_n) + S_n \sinh(X_j/L_n)} \right) \quad (3.7)$$

where n_i , N_D and N_A are the intrinsic densities, donor and acceptor density of the cell

respectively. D_p is the hole diffusion coefficient whereas D_n is the electron diffusion coefficient.

As for the recombination-dominated case, it is

$$I_0 = \frac{qAn_iW_j}{\sqrt{\tau_p\tau_n}} \quad (3.8)$$

The fill factor F.F. for a solar cell is given by

$$F.F = \frac{V_m}{V_{oc}} \left(1 - \frac{\exp(\frac{qV_m}{K_B T}) - 1}{(\exp(\frac{qV_{oc}}{K_B T}) - 1)} \right) \quad (3.9)$$

where V_m is given by the relationship as follows:

$$\exp(\frac{qV_m}{K_B T})(1 + \frac{qV_m}{K_B T}) = \frac{I_{sc}}{I_0} + 1 \quad (3.10)$$

Therefore, the conversion efficiency of a single-junction solar cell is given by

$$\eta_c = \frac{J_{sc} V_{oc} F.F.}{P_{in}} \quad (3.11)$$

where P_{in} is the incident solar power density. P_{in} for different air mass ratio is calculated from LOWTRAN6 [59].

3.3 Effects of Extrinsic Parameters on Device Modeling

In practical solar cell design, series resistance, contact materials, grid line configuration, antireflection coating and high sun insolation can affect the performance of the solar cell. Therefore, these effects must be taken into account in the optimization cycle in order to ascertain an realistically optimal design. In this section, the relationships between these extrinsic parameters and four major solar cell parameters are described.

3.3.1 Effect of Antireflection Coating

For a single layer of non-absorbing medium, the reflectivity at normal incidence of a given wavelength is given as [60]

$$R = \left(\frac{n_1^2 - n_0 n_2}{n_1^2 + n_0 n_2} \right)^2 \quad (3.12)$$

where n_0 , n_1 and n_2 are the refractive indices of air, antireflection coating material and solar cell respectively. The thickness of the antireflection coating is

$$d_{AR} = \frac{\lambda_0}{4n_1} \quad (3.13)$$

A double layer coating can offer less reflectivity loss over a wider region than a single layer coating can. A single layer has a zero reflectivity loss at only one wavelength and has an acceptable reflectivity loss over only a limited range. The reflection coefficient R of double layers is [60]

$$R = \left(\frac{n_1^2 n_3 - n_2^2 n_0}{n_1^2 n_3 + n_2^2 n_0} \right)^2 \quad (3.14)$$

where n_2 and n_3 are the refractive indices of the second coating layer and solar cell respectively and n_0 and n_1 are the same as defined in Eqn. 3.12. The thicknesses of these two coating layers can be estimated by the following relationship:

$$n_1 d_1 = n_2 d_2 = \frac{\lambda_0}{4} \quad (3.15)$$

3.3.2 Effect of Grid Design

As the area of solar cell increases, it becomes more important to include the interconnect configuration in the grid process [61]. Different grid geometry can affect the sheet resistance of the series resistance. The parallel grid line design in Fig. 3.2

is the standard grid design for a solar cell. The series resistance of this grid design is approximated as [62]

$$R_s \approx \frac{\rho_s}{12D}b^2 + \frac{b}{w} \frac{\rho_f}{3h}a^2 + \frac{b}{w}R_c \quad (3.16)$$

where D is the thickness of window layer, b is finger spacing, w is finger width, h is finger height, ρ_f is emitter resistivity, ρ_s is metal finger resistivity and R_c is the contact resistance.

It should be noted that as the width of grid element increases, the series resistance decreases and hence the electrical power loss decreases. However, this is not the only consideration. When less cell area is exposed to the photons, the short-circuit current decreases. The shadow loss is calculated as the total area of the fingers divided by the whole area of the solar cell. Therefore, there is an optimal value where the sum of these two losses is a minimum.

3.3.3 Effect of Series Resistance

When series and shunt resistance problems become important, the ratios V_m/V_{oc} and I_m/I_{sc} are reduced and the ratios calculated from Eqns. 3.8 to 3.10 should be modified. For simplicity, if shunt resistance is much higher than the series resistance, which is always true for GaAs, the open-circuit voltage V'_{oc} after additional reduction is given as

$$V'_{oc} = V_{oc} - I_{sc}R_s \quad (3.17)$$

and the reduced fill factor $F.F.'$ is given as [63]

$$F.F.' = F.F. \times \left(1 - \frac{R_s I_{sc}}{V'_{oc}}\right) \quad (3.18)$$

3.3.4 Effect of High Sun Insolation

For high sun insolation, the short-circuit current is estimated simply by multiplying the number of sun insolation by the short-circuit current obtained from one sun insolation. The open-circuit voltage of high sun insolation, V_{ocx} , is given as [64]

$$V_{ocx} = V_{oc} + \frac{nK_B T}{q} \ln(X) \quad (3.19)$$

where X is the number of sun insolation. From this equation, it appears that the conversion efficiency of high sun insolation increases with any increase of open-circuit voltage. Again, this is not always true because of the reduction of the fill factor in high sun insolation. Additionally, at high sun insolation, the effect of series resistance becomes more critical because of the high I_{sc} .

3.3.5 Effect of Irradiation

After proton or electron irradiation, the minority carrier lifetime and diffusion lengths will be degraded. The diffusion length is related to the lifetime, $L^2 = D\tau$, and it is given by the usual relations [46, 47]:

$$\frac{1}{\tau} = \frac{1}{\tau_0} + K_\tau \phi \quad (3.20)$$

$$\frac{1}{L^2} = \frac{1}{L_0^2} + K_L \phi \quad (3.21)$$

where K_τ and K_L are the minority carrier lifetime and diffusion length damage constants, respectively, and ϕ is the fluence of proton or electron.

3.4 Constrained Optimization Technique

Under the constrained optimization approach, the optimal design of a solar cell can be mathematically expressed in closed form as

$$Y = \text{Max} (f(X_i) \mid G_k \leq X_k \leq H_k, i = 1, 2, \dots, N, k = 1, 2, \dots, M) \quad (3.22)$$

where X_1, \dots, X_N are the explicit independent variables to be optimized, which might be cell thickness, doping level, etc. The implicit variables X_{N+1}, \dots, X_M are dependent functions of the explicit variables. These implicit variables might be the minority carrier diffusion length, the minority carrier lifetime or others. $f(X_i)$ is the objective function which might be the solar cell efficiency expected to be maximized, or perhaps the short-circuit current expected to be optimized. The upper and lower constraints H_k and G_k may be either constants or functions of the independent variables.

The Box complex algorithm [32, 65] is one of the most efficient constrained optimization techniques currently available. This algorithm is a sequential search technique which has proven effective in solving problems with nonlinear objective function subject to nonlinear inequality constraints. Derivatives of the objective function are not required for this algorithm. This procedure will tend to find the global optimum since the initial set of guesses is randomly scattered throughout the feasible region. The algorithm, with some minor modifications for increasing the convergence rate, proceeds as follows [65, 66]:

Step 1. Initial feasible starting K guesses, where K is at least equal to $(N + 1)$, are generated. Each guess consists of N points which are generated from random numbers and constraints for each of the independent variables:

$$\begin{aligned} X_{ij} &= G_i + r_{ij}(H_i - G_i), \\ i &= 1, 2, \dots, N, \quad j = 1, 2, \dots, K - 1 \end{aligned} \quad (3.23)$$

where r_{ij} are random numbers between 0 and 1.

Step 2. The guess points must satisfy both the explicit and implicit constraints. If a constraint is violated, the point is moved to the upper bound or lower bound of the constraint which is violated. This procedure is repeated as necessary until all the constraints are satisfied.

Step 3. The objective function is evaluated at each guess. The guess having the lowest function values is replaced by a guess which is

$$X_{ij}(\text{new}) = 1.3 \left(\bar{X}_{i,a} - X_{ij}(\text{old}) \right) + \bar{X}_{i,a}, \quad i = 1, 2, \dots, N \quad (3.24)$$

where $\bar{X}_{i,a}$ is the average of the remaining guesses and is defined by

$$\bar{X}_{i,a} = \frac{1}{K-1} \left(\sum_{j=1}^K X_{ij} - X_{ij}(\text{old}) \right), \quad i = 1, 2, \dots, N \quad (3.25)$$

Step 4. If a guess repeatedly gives the lowest function value on consecutive trials, it is moved to the average of the best and worst guesses. This minor modification would increase the convergence rate. Moreover, it will save computation time because additional constraint-violation checking is not necessary for this new guess.

Step 5. Convergence is assumed when the objective function value at each guess is within the tolerance which the user assigned. A flowchart illustrating the above procedure is given in Fig. 3.3.

3.5 Optimal Design of GaAs Single-Junction Solar Cells

The main task in this section is to optimize the efficiency of a single-junction solar cell. According to a previous study [52] and the work done by Knechtli et al. [19], GaAs single-junction solar cells with bandgap energy 1.43 eV offer some advantages – for example high η_c , radiation hardness, and relative insensitivity to temperature as compared to silicon solar cells. The optimal design of a Si solar cell is also developed here for comparison purpose.

For the GaAs cell, the design problem for maximizing the efficiency η_c is formulated as follows:

$$\eta_c = \text{Max} (f(N_D, N_A, X_j, T_j, E_g, S_n, S_p)) \quad (3.26)$$

such that

$$\ln(2 \times 10^{15} \text{cm}^{-3}) \leq \ln(N_D) \leq \ln(10^{19} \text{cm}^{-3}) \quad (3.27)$$

$$\ln(2 \times 10^{15} \text{cm}^{-3}) \leq \ln(N_A) \leq \ln(10^{19} \text{cm}^{-3}) \quad (3.28)$$

$$0.05 \mu\text{m} \leq X_j \leq 0.5 \mu\text{m} \quad (3.29)$$

$$0.1 \mu\text{m} \leq T_j \leq 10 \mu\text{m} \quad (3.30)$$

$$1.43 \text{eV} \leq E_g \leq 1.43 \text{eV} \quad (3.31)$$

$$10^4 \text{cm/s} \leq S_n \leq 10^6 \text{cm/s} \quad (3.32)$$

$$10^4 \text{cm/s} \leq S_p \leq 10^6 \text{cm/s} \quad (3.33)$$

in which the parameters to be optimized are intrinsic structural parameters (N_D , N_A , X_j) and T_j and material parameters (S_n , S_p and E_g) whose values are constants. The material parameters are determined by the materials and by the top and back contact surface passivations. For the GaAs solar cell, a thin (AlGa)As window layer is grown on the top for reducing the surface recombination velocity [19, 20]. The choices of the upper and lower bounds of the constraints are based on the considerations of contact resistance [67], the state of the art technology and radiation hardness.

Table 3.1 and 3.2 list the comparisons between the results calculated here and the published data for the optimal designs of Si and GaAs solar cells. Our designs show much better performance than those obtained from previous models [19-21, 56, 57, 68]. Moreover, they are close to the cell efficiencies made by the state of the art

[69], namely 20% for Si solar cell and 26% for GaAs solar cell respectively. The major discrepancies between our simulations and the previous models are the selections of the constraints. For instance, for an Si solar cell, N_D should be greater than 10^{19} cm^{-3} to obtain a good contact whereas N_A is two orders less than N_D [67]. In this respect, the values of N_D in a p/n junction Si solar cell can be much less than that in n/p Si solar cell and therefore results in better performance.

The effect of series resistance on GaAs solar cell efficiency is described in Table 3.3. Table 3.4 shows the optimal designs of the 300 KeV proton irradiated GaAs solar cell. It is clear that the junction thickness decreases when the proton fluence increases, and the effect of fill factor due to the irradiation is negligible. These results conform to the assumptions made for calculating the degradation of proton or electron irradiated solar cells in Chapter 2. Figure 3.4 shows the temperature and air mass ratio dependence of the efficiency of a GaAs solar cell. It shows that the efficiency of the GaAs solar cell increases as the temperature decreases. This is reasonable because the open-circuit voltage decreases when the temperature increases.

3.6 Summary

A new method of incorporating an efficient optimization algorithm into device modeling techniques for the optimal design of GaAs and Si solar cells has been examined. A modified Box complex optimization technique and a device modeling that considers the effects of extrinsic parameters has been implemented to obtain the optimal design of a GaAs single-junction solar cell. All the parameters to be optimized are adjusted in a systematic way, resulting in a truly optimal design. Comparisons between the calculated results and the published data have been made to verify the optimal designs. The optimal efficiency of the single-junction GaAs solar cell obtained by the simulation is 27.8% at room temperature for AM0 which is close to the cell efficiency, namely 26%, provided by the state of the art.

Table 3.1 Comparisons between calculated results and published data for the optimal designs of Si single-junction solar cell at room temperature.

	AM0	AM1	AM1[56]
N_D (cm ⁻³)	5.0×10^{16}	5.0×10^{16}	3.0×10^{19}
N_A (cm ⁻³)	5.0×10^{17}	1.16×10^{17}	6.0×10^{16}
X_j (μm)	0.5	0.623	0.1
T_j (μm)	300	200	66.4
S_n (cm/s)	10000	10000	118
S_p (cm/s)	10000	10000	100
V_{oc} (V)	0.89	0.88	0.63
J_{sc} A/cm ²	0.0372	0.0328	0.034
F.F.	0.85	0.85	-
η_c	21.1%	23.8%	18.06%
E_g	1.12	1.12	1.12
Collection efficiency	0.958	0.944	-
Sun Insolation	1	1	1
Shadow loss	0.018	0.018	-
Reflection loss	0.041	0.041	-
R_s (Ω)	0.473	0.473	-

Table 3.2 Comparisons between calculated results and published data for the optimal designs of GaAs single-junction solar cell at room temperature.

	AM0	AM0[68]	AM0[25]
N_D (cm^{-3})	3.16×10^{17}	2×10^{17}	7×10^{17}
N_A (cm^{-3})	2.60×10^{18}	5×10^{18}	1.5×10^{16}
X_j (μm)	0.5	0.3	0.5
T_j (μm)	10	2.5	3.0
S_n (cm/s)	10000	-	-
S_n (cm/s)	10000	-	-
V_{oc} (V)	1.23	1.05	1.04
J_{sc} A/cm^2	0.0345	0.0341	0.0317
F.F.	0.888	0.85	0.83
η_c	27.8%	22.5%	20.3%
E_g	1.43	1.43	1.43
Collection efficiency	0.999	-	-
Sun Insolation	1	1	1
Shadow loss	0.018	-	-
Reflection loss	0.041	-	-
R_s (Ω)	0.473	-	-

Table 3.3 Effect of series resistance on GaAs single-junction solar cell efficiency.

Grid Number	R_s (Ωcm^2)	η_c %
0	0	29.4
24	0.473	27.7
18	0.969	26.9
16	1.22	26.5
12	2	25.4
9	3	23.9

Table 3.4 Effect of irradiation on GaAs single-junction solar cell for 300 KeV proton at fluences of 10^{10} , 10^{11} and 10^{12} cm^{-2} .

	None	10^{10} cm^{-2}	10^{11} cm^{-2}	10^{12} cm^{-2}
$N_D (\text{cm}^{-3})$	10^{17}	10^{17}	3×10^{17}	10^{17}
$N_A (\text{cm}^{-3})$	10^{18}	10^{18}	10^{18}	10^{18}
$X_j (\mu\text{m})$	0.5	0.5	0.4	0.175
$T_j (\mu\text{m})$	10	10	10	10
$V_{oc} (\text{V})$	1.209	1.15	1.15	1.08
$J_{sc} \text{ A/cm}^2$	0.0347	0.0315	0.0283	0.0241
F.F.	0.887	0.883	0.88	0.88
η_c	27.5%	23.7%	21.3%	17.0%

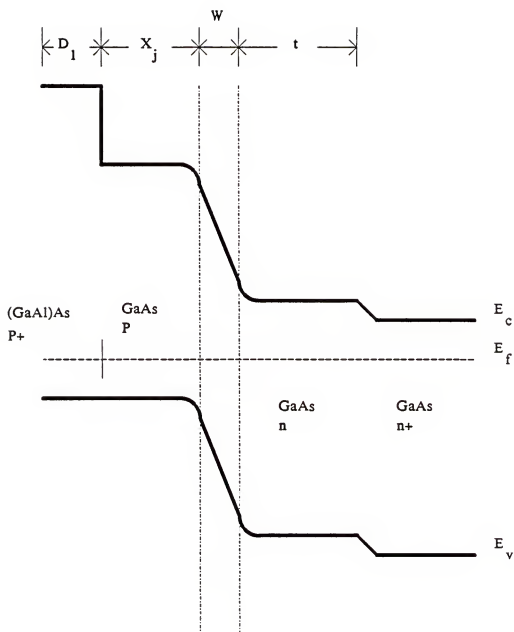


Fig. 3.1 The energy band diagram for a (AlGa)As-GaAs p-n junction solar cell.

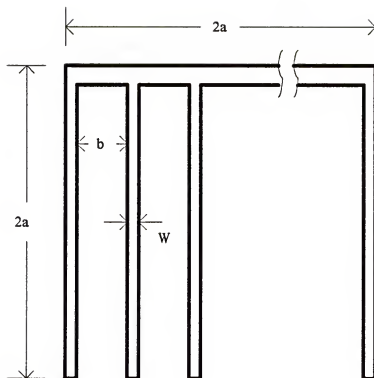


Fig. 3.2 Parallel gridline pattern for solar cells.

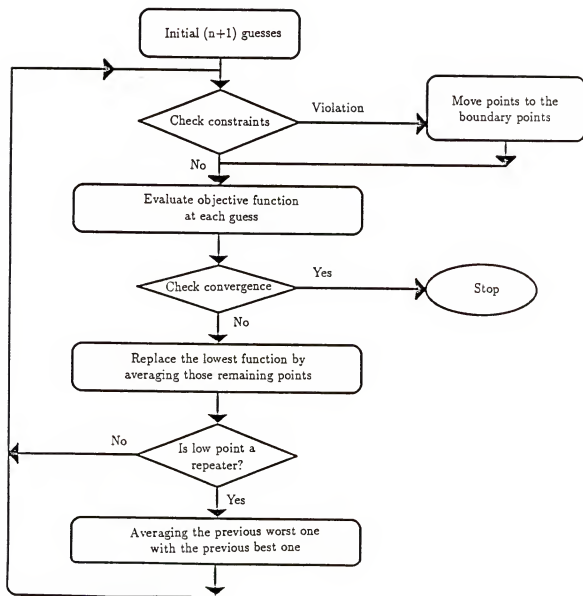


Fig. 3.3 Flowchart for the modified Box optimization algorithm.

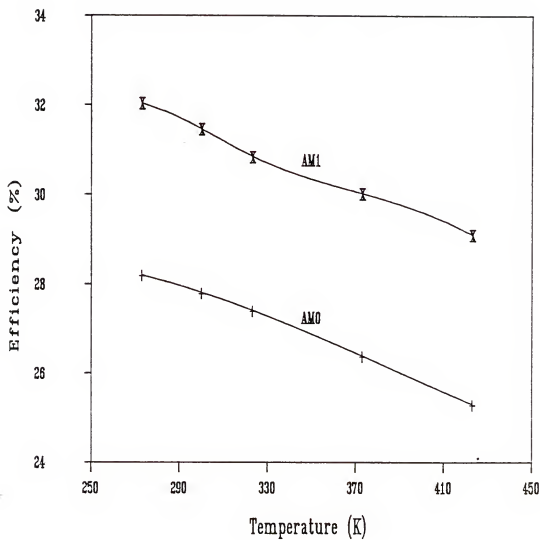


Fig. 3.4 Temperature and air mass ratio dependence of the efficiency of GaAs single-junction solar cell.

CHAPTER 4

AN EXPERT SYSTEM APPROACH TO THE OPTIMAL DESIGN OF MULTIJUNCTION SOLAR CELLS

4.1 Introduction

Single-junction solar cells today are those most frequently used. They are made from a single material with a single bandgap to absorb the sun light. The incident photon with energy less than the bandgap of the material can not be converted to electricity. And much of the photon with energy greater than the bandgap is wasted as heat. Therefore, the efficiency of a such single-junction cell is limited. In multijunction cells, solar cells with different bandgaps are put on top of each other in decreasing bandgap order. In this way, all the incident photons with energies equal to or greater than the bandgap of the top cell are absorbed by the top cell. The incident photons with energies less than the bandgap of the top are transmitted to the next cell. The phenomenon recurs at each cell.

In 1955, Jackson [3] proposed that the efficiency of solar cells could be increased significantly by constructing a system of stacked p/n homojunction photovoltaic cells which are composed of different semiconductor materials. The next year, Loferski [4] was the first to use multiple cell concepts to optimize the efficiency of a photovoltaic system, selecting sets of cells which made use of the energy available from the entire solar spectrum. Recently, many researchers [3-18] have proposed varying approaches to the optimal design of the multijunction cells both theoretically and experimentally. However, their results were not realistic for several reasons. First, data for the electrical and optical parameters of the materials they used were not always available and accurate. Secondly, their designs were based on either a small set of design parameters or an inadequate device modeling. Thirdly, they did not

consider the interrelation of the parameters of the solar cells and hence their results were not optimal.

In this chapter, we propose for the first time an expert system approach to the optimal design of multijunction solar cells for both terrestrial and space applications. An expert system is a knowledge-intensive computer program. The knowledge of an expert system consists of facts and rules. The facts constitute a body of information that is widely shared, publicly available and generally agreed upon by experts in the field [33, 34, 70]. The rules are those if-then rules that characterize expert level decision making in the field. In general, an expert system with a large number of facts and rules is better than one with just a few facts and rules. To assure the usefulness of our expert system, we have implemented as many facts and rules as possible with the help of published literatures and experts in the photovoltaic area. However, because of the availability of other research reports and the tradeoff between the number of rules and the computation time, our expert system is based on the AlGaAs, GaAs, $\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$, Si and Ge materials. And less than a hundred rules are actually implemented.

Section 4.2 deals with the device modeling of multijunction tandem solar cells used in the expert system. In section 4.3, the concept of an expert system approach to the optimal design of multijunction solar cells is presented. An optimization technique with heuristic rules is also shown in this section. Results and discussion are in section 4.4, with a summary in section 4.5.

4.2 Device Modeling of Multijunction Tandem Solar Cells

Since the theoretical conversion efficiency increases very slowly beyond three-junction tandem cells [1] and the technology of fabricating cells beyond three junction is difficult in practice [2], the multijunction study focused on those structures that are two-junction and three-junction only. In general, the calculations of I_{sc} , V_{oc} , F.F. and η_c for the multijunction cells are similar to those for single-junction solar cell

presented in Chapter 3. Here, we will only illustrate the three-junction calculations.

Let the top, middle and bottom cells have the bandgaps of E_{g1} , E_{g2} and E_{g3} with corresponding short-circuit currents I_{sc1} , I_{sc2} and I_{sc3} respectively. Then, the short-circuit currents can be written as

$$I_{sc1} = \int_{0.3}^{\lambda_1} I_{E1}(\lambda) + I_{B1}(\lambda) + I_{W1}(\lambda) d\lambda \quad (4.1)$$

$$I_{sc2} = \int_{\lambda_1}^{\lambda_2} I_{E2}(\lambda) + I_{B2}(\lambda) + I_{W2}(\lambda) d\lambda \quad (4.2)$$

$$I_{sc3} = \int_{\lambda_2}^{\lambda_3} I_{E3}(\lambda) + I_{B3}(\lambda) + I_{W3}(\lambda) d\lambda \quad (4.3)$$

where λ_1 (μm) = $1.24 / E_{g1}$, λ_2 (μm) = $1.24 / E_{g2}$, and λ_3 (μm) = $0.24 / E_{g3}$ respectively. The current components from emitter, base and space charge of each cell and other notations in Eqn. 4.1 to Eqn. 4.3 are similar to those given for the single-junction solar cell.

The representations of the open-circuit voltage, fill factor and conversion efficiency of each cell are also similar to Eqns. 3.6, 3.9 and 3.11 respectively. For the two-terminal case as shown in Fig. 4.1, the I_{sc} values are all the same for the top, middle and bottom cells, namely the smallest value among the three cells. The total conversion efficiency of the three-junction cells is then

$$\begin{aligned} \eta_{tot} &= \eta_{c1} + \eta_{c2} + \eta_{c3} \\ &= \frac{I_{sc} V_{oc1} F.F._1 + I_{sc} V_{oc2} F.F._2 + I_{sc} V_{oc3} F.F._2}{P_{in}} \end{aligned} \quad (4.4)$$

Those effects such as series resistance, grid structures, etc., which have been discussed in Chapter 3, are still applicable to the design of multijunction solar cells. They are not repeated here. However, the effect of the tunnel diode should be considered. The role of a tunnel diode in the design of the multijunction solar cell is to connect the two

different p/n junctions. The doping density of the tunnel diode must be quite high in order to have a low impedance to current flow in both directions and the voltage drop across it should be as small as possible [5]. Additionally, the bandgap of the tunnel diode should be as large or larger than the top cell bandgap so that as many photons as possible can be transmitted from the top cell and absorbed at bottom. According to Kane's theory, the tunnel current is given as [71, 72, 73]

$$\frac{J_t}{V} = \frac{q^2 \sqrt{2m^* E_g}}{4\hbar^2 W_t \pi^3} \exp\left(-\frac{\pi \sqrt{m^* E_g} W_t}{2\sqrt{2}\hbar}\right) \quad (4.5)$$

where m^* is the effective mass of the tunnel diode; W_t is the depletion width of the tunnel diode; E_g is the bandgap of the tunnel material; V is the voltage across the tunnel diode and J_t is the tunnel current density. Therefore, the open-circuit voltage of the solar cell should be adjusted by the amount of the voltage across the tunnel diode.

An alternative approach for the intercell ohmic contact of the multijunction solar cells is the metal intercell contact shown in Fig. 4.2. In the metal interconnect contact technique, the top (AlGa)As cell and middle cell will be interconnected by the metal interconnect technique. Grooves will be etched through the top cell to reach the middle cell. Metals will be deposited within the grooves to shorten the base region of the top cell to the emitter layer of the middle cell. The connection between the middle and the bottom cell is made by fabricating a tunnel junction as shown in Fig. 4.3. The metal intercell contact structure has the problems of a complex fabrication process and double shadowing. The total series resistance of the multijunction cells is then the sum of the series resistances of top cell, bottom cell and emitter layer of the middle layer.

4.3 Concept of the Expert System Approach

The shortcomings of the conventional computer programs for the optimum de-

sign of high conversion efficiency single-junction and/or multijunction solar cells are twofold. First, they are not flexible: due to the intermixture of data and codes, the programs are not always applicable without code corrections when the goal of the design changes. Second, results obtained from those programs are sometimes unrealistic: data or design criteria often are not properly taken into account.

In this section, we propose a new approach to formulating the design problem and then develop an expert system to aiding the optimal design of high efficiency single-junction and/or multijunction solar cells for both space and terrestrial applications. Our proposed expert system will apply to the radiation free environment as well as to the proton or electron irradiation environment. A systematic approach to this design problem consists of four tasks (1) constructing a solar cell knowledge data base, (2) searching for all possible designs from the knowledge base, (3) selecting the optimum design by applying heuristic rules, and (4) verifying the optimum design. This expert system will produce practical designs corresponding to user specifications and thus become an important tool for the solar cell designer.

4.3.1 Problem Formulation

An expert system approach to solving for the optimal design of multijunction solar cells will usually be successful if the theoretical rules and/or experienced rules have been formulated quantitatively. The design procedures of the proposed expert system can be formulated into four tasks. The first task is the construction of a solar cell knowledge base. Here the first step is to define the problem domain precisely. This will include an analysis of the desired results from the expert system. Our expert system is currently limited to (AlGa)As, GaAs, Si, Ge and $\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$ materials. The goal of the proposed solar cell expert system is to optimize the conversion efficiency of a 2-terminal monolithic single-junction or multijunction solar cell based on these five material systems. Then the types of knowledge required must be identified as well as the possible interactive consultation and the approximate number of rules,

which will reduce the search space of the design problem. In this respect, the facts in the solar cell knowledge base will be the fundamental properties of the materials such as the bandgap, minority carrier diffusion length, minority carrier lifetime, contact resistance, etc. The data structure of these facts can be represented as lists, trees, nets, rules or other formations. Since most of the fundamental properties, namely facts, are a function of doping density, temperature etc., the if-then rules representation for the facts is used for the expert system.

The rules consist of antecedent-consequent pairs. The characteristic of a rules-based system is the separation of data examination from data modification. The examination of data generally occurs in the antecedent of a rule, while data modification is normally managed by the consequent of a rule [70]. For example, if a GaAs p/n junction solar cell is chosen, then the junction thickness of the cell should be less than $0.5 \mu\text{m}$ in order to obtain high efficiency and high radiation hardness. By the application of this rule, the search space of the problem is reduced and computation time is saved. However, there may be a number of conflict rules in the knowledge base and a selection must be made. Therefore, a conflict resolution process is set up to determine which satisfied rule to be used in the conflict set.

There are two ways of constructing the rules-based system of the expert system. The first method is to let the expert system automatically adopt the solar cell properties from existing data files or tables. These data may be the results obtained from the expert system itself. The other way is through a set of queries under a certain condition C as

$$C = (C_1, C_2, \dots, C_n) \quad (4.6)$$

where C_j is a value under condition j , and n is the total number of queries. The queries might be of the following type:

1. Input the cell name please.

2. Input the bandgap of this cell please.
3. Input the contact material please.

The second task is to search the knowledge data base. Let G_i be the mapping function from a description of a solar cell to the value of the material with respect to a property i under a certain condition C . A solar cell is described by a set of tuples, e.g. bandgap energy, short-circuit current, open-circuit voltage, conversion efficiency and so on, as

$$M_l = (m_1, m_2, \dots, m_k) \quad (4.7)$$

where l is the name of a solar cell and k is the total number of tuples. The retrieved value V_{ki} of a solar cell M_l with regard to property i is represented by G_i . And

$$V_{ki} = G_i(M_k) \quad (4.8)$$

The set of solar cells which satisfy the requirements on property i is

$$S_i = \{s | C_i \leq G_i(s), s \in M\} \quad (4.9)$$

where M is the universal set of M_j ($j = 1, 2, \dots$) created in the knowledge data base. The solar cell which satisfies the requirements are obtained by the intersection of the sets as

$$D = S_1 \cap S_2 \cap S_3 \dots \cap S_r \quad (4.10)$$

However, the set D of the selected solar cells can be empty in some cases, and then the procedures for constructing the knowledge data base must be invoked.

The third task is the selection of the optimal designs. The selection of optimal design D_{opt} is performed by calculating the scalar product of a weighting factor $W = (w_1, w_2, \dots, w_r)$ predetermined by the solar cell designer and a matrix V_s as

$$E(e_1, e_2, \dots, e_n) = WV_s \quad (4.11)$$

where n is the possible designs and e_i is a value of this evaluation for the selected solar cell i . The weighting factors can be the cost of solar cells, complexity of fabrication processes, conversion efficiency, weight of solar cells, preference of materials and radiation hardness. In this expert system, it is assumed that the weighting factors of the radiation hardness and conversion efficiency are dominant.

The last task is to verify the optimal design. The verification of the optimal design D_{opt} is made by comparing the calculated results obtained from this expert system with the experimental and/or theoretical data collected from the literature of the photovoltaic field.

4.3.2 Optimization and Heuristic Rules

For the multijunction solar cells, the optimization problem, which is to maximize the total efficiency of each subcell, can be formulated as

$$\eta_{tot} = \sum_{i=1}^n \eta_{ci} \quad (4.12)$$

where n is the number of the subcells of the multijunction solar cell. And

$$\eta_{ci} = \text{Max}(f(N_{Di}, N_{Ai}, X_{ji}, T_{ji}, E_{gi}, S_{ni}, S_{pi})) \quad (4.13)$$

It should be noted that the format of Eqn. 4.13 is similar to that of Eqn. 3. 26. Consequently, according to Eqn. 4.12 and 4.13 the number of constraints for the optimal designs of two-junction and three-junction solar cells will be two times

and three times more than those of a single-junction solar cell. For example, a three-junction solar cell needs at least 15 constraints if S_n and S_p are fixed. Since the computation time will be increased exponentially instead of linearly as the number of the constraints increase, a couple of heuristic rules have been adopted in this system in order to prune the search space and hence to save the computation time.

A heuristic is a technique that improves the efficiency of the search process and leads to an adequate answer, if not the best one of a difficult problem [74]. A heuristic rule serves as an aid to problem solving by experimental, especially trial-and-error, methods [75]. Therefore, it is possible to construct a special purpose heuristic rule that exploits domain-specific knowledge to solve a particular problem. In this respect, heuristic rules are applied to the selections of the upper bound and/or lower bound of the constraints, and to the combinations of material in the system for multijunction solar cells, in this way to obtaining a quick feasible optimal solution. Then, the solution will be compared with the existing data to justify the validity of heuristic rules.

4.4 Results and Discussion

The design of the multijunction tandem cells is considerably more complex than that of single-junction cells and hence additional parameters must be considered. They are as follows [76, 77]:

1. Bandgap energies must be optimized for multijunction solar cells.
2. Lattice matching is desired.
3. Direct optical transitions are desirable.
4. Metallurgical system must be compatible.
5. A compatible substrate must be available.

6. Must be invariant with changes in the environment.

According to our computer calculations, the optimum bandgap combinations of two-junction and three-junction tandem cells are 1.75/1.10 eV , 2.02/1.43 eV, 2.00/1.43/1.04 eV and 2.00/1.40/1.00 eV under AM0 at room temperature. However, considering the lattice matching problem, it is clear that the materials selected for the top and middle cells, (AlGa)As and GaAs are quite favorably lattice matched for the two-junction tandem cells. Although the (AlGa)As/Si and (AlGa)As/(InGa)As two-junction solar cells meet the optimum bandgap energy combinations, they are still not the qualified designs due to the lattice constants and thermal expansion coefficients mismatch. In addition, the radiation hardness of GaAs is greater than that of Si or (InGa)As.

As for the three-junction structure, (InGa)As, Ge and Si can be the possible candidate cells for the bottom cell. However, these materials exhibit lattice mismatch with respect to GaAs and generate and propagate dislocations which may adversely affect solar cell performance.

In a multijunction solar cell it is advantageous to have direct bandgap materials, to reduce the thickness of the required material. This reduction results not only in lower material costs but also in lower epitaxial growth costs for growing the required layers. Obviously, it is also beneficial to minimize thickness to gain a weight advantage for the final structure. Even more significant, however, is the reduction in growth time. In addition, the thinner layers inherent in the direct bandgaps materials tend to lead to lower minority carrier recombination losses and improved radiation hardness. Therefore, the top cell for our optimal design of the GaAs based multijunction solar cell is limited to the direct bandgap material.

A comprehensive understanding of the electronic transport properties of the materials and of the total assembly in the two-junction or three-junction tandem cells is essential to a complete analysis of the relative merits of the final choices. The most

tractable parameter which reflects the quality of the device is probably the minority carrier diffusion length. A long diffusion length would require that the defects and recombination centers be minimized, that the junction quality be kept as high as possible and that the interfaces between layers be kept free of strain and imperfections. Since (AlGa)As, GaAs and (InGa)As are polar materials which are different from the nonpolar material such as Ge and Si, a characterization of the dominant scattering mechanisms that affect the mobility and hence affect the diffusion length should be done to facilitate the study of the optimal designs. Detailed discussion of different scattering mechanisms and calculations of the electron and hole mobilities of the (AlGa)As is given in Chapter 5.

Table 4.1 and Table 4.2 list the results of our calculated two-junction and three-junction solar cells respectively. For those two-junction solar cells except GaAs/Ge in Table 4.1, an $\text{Al}_x\text{Ga}_{1-x}\text{As}$ heavy-doped tunnel junction with Al composition greater than 0.45 has been used for our simulations. However, a heavy-doped GaAs tunnel junction is used for the case of GaAs/Ge. For the three-junction solar cells in Table 4.2, an (AlGa)As tunnel junction is used for connecting the top cell and middle cell whereas a GaAs tunnel junction is used for connecting the middle cell and bottom cell. Here, we assume that it is feasible to obtain an (AlGa)As tunnel junction which is as good as the GaAs tunnel junction. According to the state of the art technology [78, 79], the n-type and p-type doping densities of GaAs can be as high as $2 \times 10^{19} \text{ cm}^{-3}$ and $5 \times 10^{19} \text{ cm}^{-3}$ respectively. The effect of doping densities of the tunnel junction on the efficiency of a multijunction solar cell is shown in Table 4.3. It is found that the voltage drop across the tunnel junction decreases as the doping densities increase and hence the performance of the solar cell increases. An alternative approach is to use the MIC structure to shorten the top and bottom cell for a two-junction solar cell. However, according to our simulation results, the efficiency of the (AlGa)As/GaAs two-junction solar cell obtained from this method, for example 24%, is much lower

than that in Table 4.1. This inferiority is due to the increases of the shadowing loss and series resistance. It may be desirable for the optimal designs of multijunction solar cells only if the high quality tunnel junction is not feasible. The optimal designs of two-junction and three-junction solar cells are listed in Table 4.4 and Table 4.5 respectively. The temperature and air mass ratio dependence of the efficiency of the $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}/\text{GaAs}$ and $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}/\text{GaAs}/\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$ are shown in Fig. 4.4 and Fig. 4.5.

4.5 Summary

An expert system approach to the optimal design of two-junction and three-junction solar cells has been presented in this chapter for the first time. A rule-based system with a couple of heuristic rules is implemented in the expert system to prune the search space of the design problem and hence to save the computation time. The optimal designs of $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}/\text{GaAs}$ two-junction solar cell and $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}/\text{GaAs}/\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$ three-junction solar cell with room temperature efficiencies for AM0 of 30.01% and 35.3% respectively were obtained in the simulation.

Table 4.1 Simulation results of two-junction solar cells at room temperature for AM0.

	(AlGa)As/GaAs	GaAs/Ge	(AlGa)As/(InGa)As	(AlGa)As/Si
V_{oc1} (V)	1.52	1.22	1.24	1.44
$F.F._1$	0.91	0.895	0.896	0.91
$\eta_{c1}(\%)$	16.86	22.3	23.5	17.5
E_{g1} (eV)	1.97	1.40	1.52	1.85
V_{oc2} (V)	1.21	0.531	0.55	0.88
$F.F._2$	0.895	0.80	0.80	0.87
$\eta_{c2}(\%)$	13.1	8.68	9.37	10.3
E_{g2} (eV)	1.40	0.66	0.744	1.12
$J_{sc}(A/cm^2)$	0.0164	0.0275	0.0284	0.0180
Total $\eta_c(\%)$	30.01	31.0	32.9	27.8

Table 4.2 Simulation results of three-junction solar cells at room temperature for AM0.

	(AlGa)As/GaAs/(InGa)As	(AlGa)As/GaAs/Ge
V_{oc1} (V)	1.50	1.46
$F.F._1$	0.91	0.91
$\eta_{c1}(\%)$	16.7	15.85
E_{g1} (eV)	1.98	1.96
V_{oc2} (V)	1.21	1.21
$F.F._2$	0.896	0.896
$\eta_{c2}(\%)$	13.2	12.9
E_{g2} (eV)	1.40	1.40
V_{oc3} (V)	0.54	0.521
$F.F._3$	0.81	0.806
$\eta_{c3}(\%)$	5.34	4.99
E_{g3} (eV)	0.744	0.66
$J_{sc}(A/cm^2)$	0.0164	0.0160
Total $\eta_c(\%)$	35.3	33.78

Table 4.3 The effect of doping densities of the tunnel junction on the efficiencies of two-junction and three-junction solar cells.

Doping density	$\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}/\text{GaAs}$	$\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}/\text{GaAs}/\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$
$N_D = 2 \times 10^{19}$ $N_A = 5 \times 10^{19}$	30.0%	35.3 %
$N_D = 10^{19}$ $N_A = 5 \times 10^{19}$	27.9%	30.06 %
$N_D = 10^{19}$ $N_A = 3 \times 10^{19}$	24.6%	24.54 %

Table 4.4 Optimal design of $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}/\text{GaAs}$ two-junction solar cell at room temperature for AM0.

	$\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}$	GaAs
$N_A \text{ (cm}^{-3}\text{)}$	8.54×10^{15}	1.05×10^{18}
$N_D \text{ (cm}^{-3}\text{)}$	1.17×10^{15}	1.65×10^{17}
$X_j \text{ (}\mu\text{m)}$	0.05	0.477
$T_j \text{ (}\mu\text{m)}$	4.5	7.28
$V_{oc} \text{ (V)}$	1.52	1.21
F.F.	0.912	0.895
$\eta_c \text{ (\%)}$	16.86	13.16
$J_{sc} \text{ (A/cm}^2\text{)}$	0.0164	0.0164
$E_g \text{ (eV)}$	1.975	1.406

Table 4.5 Optimal design of $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}/\text{GaAs}/\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$ three-junction solar cell at room temperature for AM0.

	$\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}$	GaAs	$\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$
$N_A \text{ (cm}^{-3}\text{)}$	3.33×10^{15}	10^{18}	2×10^{18}
$N_D \text{ (cm}^{-3}\text{)}$	7.94×10^{14}	1.8×10^{17}	5×10^{17}
$X_j \text{ (}\mu\text{m)}$	0.05	0.5	0.36
$T_j \text{ (}\mu\text{m)}$	7.1	10	2.0
$V_{oc} \text{ (V)}$	1.50	1.21	0.541
F.F.	0.912	0.895	0.81
$\eta_c \text{ (%)}$	16.76	13.24	5.34
$J_{sc} \text{ (A/cm}^2\text{)}$	0.0164	0.0164	0.0164
$E_g \text{ (eV)}$	1.98	1.406	0.744

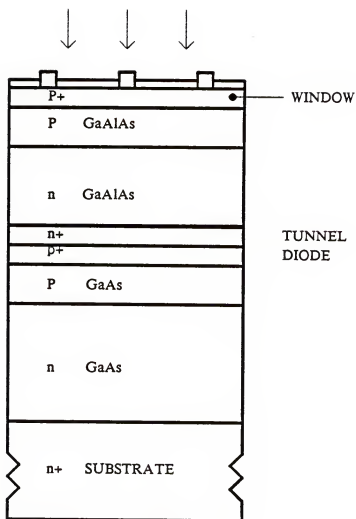


Fig. 4.1 The schematic diagram of an AlGaAs/GaAs two-junction solar cell.

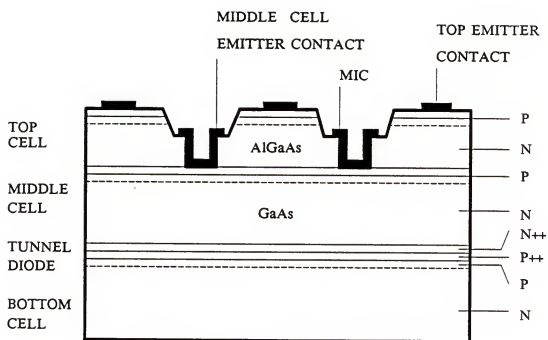


Fig. 4.2 The schematic diagram of an AlGaAs/GaAs/InGaAs three-junction solar cell.

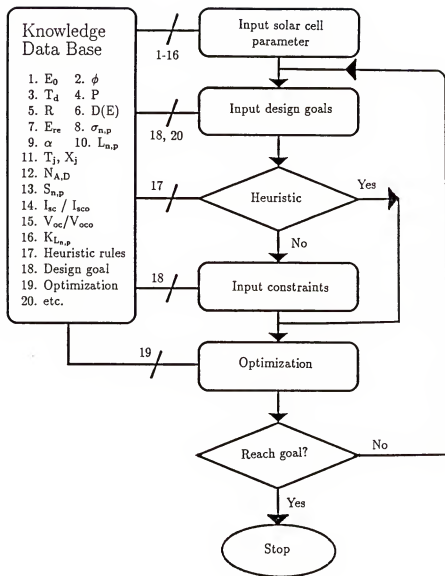


Fig. 4.3 Flowchart for the optimization of multijunction solar cells.

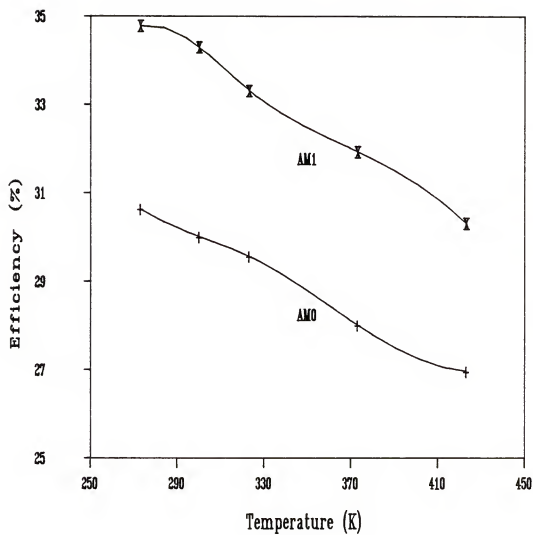


Fig. 4.4 Temperature and air mass ratio dependence of the efficiency of the AlGaAs/GaAs two-junction solar cell.

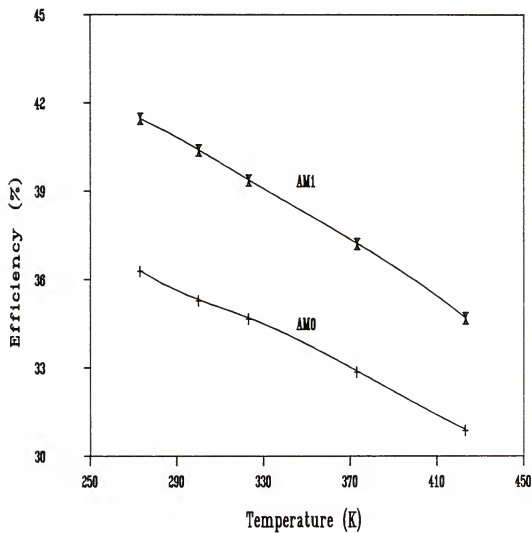


Fig. 4.5 Temperature and air mass ratio dependence of the efficiency of the AlGaAs/GaAs/InGaAs three-junction solar cell.

CHAPTER 5

THEORETICAL CALCULATIONS OF ELECTRON AND HOLE MOBILITIES IN $\text{Al}_x\text{Ga}_{1-x}\text{As}$

5.1 Introduction

Although there exists some experimental and theoretical data for electron and hole mobilities in $\text{Al}_x\text{Ga}_{1-x}\text{As}$ [80-100], the absence of a well developed experimental characterization of these mobilities as a function of Al composition, dopant density and temperature makes the development of an accurate model difficult. In this chapter, we examine various scattering mechanisms of the $\text{Al}_x\text{Ga}_{1-x}\text{As}$ and develop a good approximation for numerical simulations of these mobilities. The reasons why we probe the $\text{Al}_x\text{Ga}_{1-x}\text{As}$ alloy system particularly are twofold. First, $\text{Al}_x\text{Ga}_{1-x}\text{As}$ material is the leading candidate for the top cell of the multijunction solar cells because of its bandgap and lattice matching to the GaAs solar cell. Second, unlike the GaAs, there are few reports on high efficiency $\text{Al}_x\text{Ga}_{1-x}\text{As}$ solar cells. We believe that this is due to the low mobilities and lifetimes of $\text{Al}_x\text{Ga}_{1-x}\text{As}$. Therefore, the characterization of the dominant scattering processes that affect the electron or hole mobility and hence affect the diffusion length of $\text{Al}_x\text{Ga}_{1-x}\text{As}$ are needed to facilitate the optimal design of a GaAs based multijunction solar cell.

Instead of doing a linear extrapolation of the experimental mobility data of GaAs and AlAs, which would not be accurate at all, we investigate in detail all the possible scattering mechanisms of $\text{Al}_x\text{Ga}_{1-x}\text{As}$, thus attaining an accurate model for numerical simulation. The modeling of electron mobility is discussed first; it is more complicate than that of hole mobility due to the different bands involved.

For a full analysis of the Hall electron mobilities in the $\text{Al}_x\text{Ga}_{1-x}\text{As}$ ternary compound, we first consider all the electrons involved in the conduction process such as

those in the Γ , X and L minima. It is known that the mobilities in different scattering mechanisms – for example polar optical scattering μ_{po} , piezoelectric scattering μ_{pe} , deformation potential scattering μ_{dp} , ionized impurity scattering μ_{ii} , space charge scattering μ_{sc} , alloy scattering μ_a and intervalley scattering μ_{iv} – have all been observed in the $\text{Al}_x\text{Ga}_{1-x}\text{As}$ materials at room temperature [80-83]. According to our calculations, μ_{ii} and μ_{sc} are dominant in limiting the electron mobility at temperatures as low as 100 K in the n-type $\text{Al}_x\text{Ga}_{1-x}\text{As}$ with doping concentrations greater than 10^{18} cm^{-3} . When the temperature increases, μ_{po} and μ_{iv} become more dominant than other scatterings. As the temperature increases above room temperature, μ_{po} and μ_{iv} play the dominant role in the hall electron mobility of $\text{Al}_x\text{Ga}_{1-x}\text{As}$. It should be noted that although the model can be applied to all the different Al compositions from 0 to 1, we only show the results for Al composition between 0.10 and 0.45. This is because that bandgap region of the $\text{Al}_x\text{Ga}_{1-x}\text{As}$ is probably the potential top cell for the GaAs based multijunction solar cells.

As for the hole mobility calculations, the intervalley scattering is not considered since different bands are not involved. Our calculations show that μ_{dp} , μ_{ii} and μ_a are dominant in limiting the hole mobility in the p-type $\text{Al}_x\text{Ga}_{1-x}\text{As}$ with doping concentrations greater than 10^{18} cm^{-3} at 100 K. As the temperature increases, μ_{po} becomes more and more dominant in limiting the hole mobility of $\text{Al}_x\text{Ga}_{1-x}\text{As}$.

5.2 Scattering Processes Of $\text{Al}_x\text{Ga}_{1-x}\text{As}$

5.2.1 Polar Optical Scattering

In polar semiconductors such as III-V compounds, the interaction of carriers with the optical mode of lattice vibrations is known as polar optical scattering [84]. Because of the strong dipole moment set up by the optical modes in the polar crystals, the coupling between an electron or hole and the optical modes is likely to be much stronger than in non-optical crystals. The temperature dependence of the electron or hole mobility due to the polar optical scattering follows the analysis of Fortini et al.

[85] and is given by

$$\mu_{po} = \frac{25.54[\exp(z) - 1]G(z)T^{1/2}(m^*/m_0)^{-3/2}}{T_o(\epsilon_h^{-1} - \epsilon_l^{-1})} \quad (5.1)$$

where $z = T_o/T$ with T_o , being the longitudinal optical phonon temperature, and the function $G(z)$ was obtained by [85]. T_o is denoted by $\hbar\omega_{LO}/K_B$, where ω_{LO} is the longitudinal phonon energy and m^* is the effective mass of an electron or hole. The ϵ_l and ϵ_h are the low and high frequency dielectric constants, respectively. They are given by [86-87]

$$\epsilon_l = 10.06x + 12.91(1 - x) \quad (5.2)$$

$$\epsilon_h = 8.16x + 10.91(1 - x) \quad (5.3)$$

It should be noted that one of the crucial parameters needed for the calculations of electron mobility in the various minima for different values of x is the effective mass. The masses used in the polar optical scattering and others scatterings are the conductivity effective masses. They are given as [88]

$$m_{\Gamma}^c = 0.067 + 0.083x \quad (5.4)$$

$$m_X^c = 0.32 - 0.06x \quad (5.5)$$

$$m_L^c = 0.11 + 0.3x \quad (5.6)$$

Since the hole mobility does not involve different bands transitions, only one hole mobility effective mass is needed for the calculations. It is defined as [72]

$$\frac{1}{m_h^*} = \frac{\sqrt{m_{lh}} + \sqrt{m_{hh}}}{m_{lh}\sqrt{m_{lh}} + m_{hh}\sqrt{m_{hh}}} \quad (5.7)$$

where m_{lh} and m_{hh} are the light hole and heavy hole effective masses. They are given as [88]

$$m_{lh} = 0.087 + 0.063x \quad (5.8)$$

$$m_{hh} = 0.62 + 0.14x \quad (5.9)$$

5.2.2 Piezoelectric Scattering

If a III-V compound semiconductor consists of dissimilar atoms such as $Al_xGa_{1-x}As$ where the bonds are partly ionic and the unit cell does not contain a center of symmetry, carriers may be scattered by longitudinal acoustic waves due to piezoelectric scattering. Since the lattice constant of $Al_xGa_{1-x}As$ material is almost independent of Al composition, its elastic constants should be nearly the same as for GaAs. With this assumption, μ_{pe} is given as [89]

$$\mu_{pe} = 43 \frac{\sqrt{300}}{T} \frac{\epsilon_1}{(m^*/m_0)^{3/2}} \quad (5.10)$$

According to our calculation μ_{pe} is negligible. It should be noted that all the parameters with same names in these different scattering processes are defined as the same unless specified otherwise.

5.2.3 Deformation Potential Scattering

The scattering of an electron or hole by the longitudinal acoustical phonon is an important scattering for many semiconductors near room temperature. The scattering is elastic if the electron or hole energy is much greater than the phonon energy and the change in electron or hole energy during scattering is small compared to the average energy of an electron or hole. The μ_{dp} due to the acoustical mode scattering has been derived by Bardeen and Shockley as [90]

$$\mu_{dp} = 3.17 \times 10^{-5} \frac{\rho u_\ell^2 T^{-3/2}}{E_1^2 (m^*/m_0)^{5/2}} \quad (5.11)$$

where the mass density $\rho = 5.37 (1 - x) + 3.60 x$ [86, 91] and u_ℓ is the longitudinal sound velocity. The deformation potential E_1 is equal to $(6.7 - 1.2x)$ [88].

5.2.4 Ionized Impurity Scattering

The scattering of an electron or hole by an ionized impurity center in a semiconductor is an example of elastic scattering. This is due to the fact that the mass of an impurity atom is much greater than that of an electron or hole. μ_{ii} can be calculated from the expression given by [92]

$$\mu_{ii} = \frac{64\sqrt{\pi}\epsilon_0^2\epsilon_s^2(2K_B T)^{3/2}}{N_I q^3 \sqrt{m^*} \ln\left(\frac{24m^*\epsilon_0\epsilon_s K_B^2 T^2}{q^2 h^2 N_I}\right)} \quad (5.12)$$

where N_I is the ionized impurity density, and h , K_B and ϵ_0 are the Planck constant, Boltzmann constant and permittivity in vacuum, respectively.

5.2.5 Space Charge Scattering

The space charge scattering is caused by the crystal inhomogeneities. Such inhomogeneities may act like a mobility killer center and probably result from the grown-in defects. Weisburg [93] and Conwell and Vassel [94] have derived the expression for the μ_{sc} due to the space charge scattering which is

$$\mu_{sc} = \frac{3.2 \times 10^9}{N_s \sigma \sqrt{T m^*/m_0}} \quad (5.13)$$

where N_s and σ are the density and cross section of the space charge scattering centers. Saxena and Mudares [81] showed that the best fitted value of $N_s \sigma$ was $9.5 \times 10^4 \text{ cm}^{-1}$, whereas Stringfellow [83] showed that $N_s \sigma$ was a function of the alloy composition x , which is given by

$$N_s\sigma = 5 \times 10^3 + 6.3 \times 10^5 x \quad (5.14)$$

However, according to our calculations the optimal fitted value is $4.5 \times 10^4 \text{ cm}^{-1}$. This is in close agreement with the capture cross sections and defect densities obtained from our DLTS measurements.

5.2.6 Alloy Scattering

In the III-V ternary compound semiconductors of the type $A_xB_{1-x}C$, the constituent elements A and B are randomly distributed among the C atoms. This random distribution constitutes a random potential component to the periodic potential, which causes an additional scattering process known as the alloy scattering. The alloy scattering mobility μ_a is given by [89, 95, 96]

$$\mu_a = 52.3T^{-1/2} \{ (m^*/m_0)^{5/2} x(1-x)(E_a)^2 \}^{-1} \quad (5.15)$$

where $E_a = 0.3 + 0.011x$ [89] is the alloy scattering potential.

5.2.7 Intervalley Scattering

Since there is no band to band transition involved for hole, the intervalley scattering is for electron only. The scattering rate from a k state in the i valley to a state in the j valley has been derived by Fawcett et al. [97], the mobility limited by nonequivalent intervalley scattering can be expressed as [89, 98]

$$\mu_{iv} = \frac{4\sqrt{2/3}\mu_{dp}}{Z_j(T_c/T)^{3/2}} \left(\frac{[T/T_c + 2/3 + (2/3)(\Delta E/K_B T_c)]^{1/2}}{\exp(T_c/T) - 1} + \frac{[T/T_c - 2/3 - (2/3)(\Delta E/K_B T_c)]^{1/2}}{1 - \exp(-T_c/T)} \right)^{-1} \quad (5.16)$$

where ΔE is the subband gap among the minima involved in the process and Z_j is

the number of nonequivalent intervalley. For equivalent intervalley scattering, the Z_j is changed to $(Z_j - 1)$ and $\Delta E = 0$. Since the masses of group V atoms are larger than those of group III atoms in $\text{Al}_x\text{Ga}_{1-x}\text{As}$, we have selected the longitudinal optical phonon involved in the process of scattering among the L and X minima, and hence $T_c = T_o$ can be assumed [99]. It should be noted that because there is only one Γ band minimum, there is no equivalent intervalley scattering in the Γ band.

5.3 Analysis and Discussion

To analyze the mobility data, the following assumptions are made : (a) the electrons or holes are scattered in a parabolic band, (b) the various scattering mechanisms are independent of each other, and (c) Matthiessen's rule for calculating the electron or hole mobility in $\text{Al}_x\text{Ga}_{1-x}\text{As}$ is valid. The Γ , X, L conduction band structure as a function of alloy composition plays an important role in determining the electron mobility. For the electron mobility in the composition range $0 < x < 0.32$, the transport properties are primarily determined by the electrons in the Γ conduction minimum, and the effects of the L and X minima may be negligible. Similarly, for x is greater than 0.6, the X minima plays the major role. In the intermediate band crossover composition range, i.e., for $0.32 < x < 0.6$, the effects of Γ , L and X minima must be taken into account. Considering the three-valley conduction, n_H and μ_H can be expressed as [100]

$$n_H = n_\Gamma \frac{(1 + (n_X \mu_X)/(n_\Gamma \mu_\Gamma) + (n_L \mu_L)/(n_\Gamma \mu_\Gamma))^2}{1 + (n_X/n_\Gamma)(\mu_X/\mu_\Gamma)^2 + (n_L/n_\Gamma)(\mu_L/\mu_\Gamma)^2} \quad (5.17)$$

and

$$\mu_H = \mu_\Gamma \frac{1 + (n_X/n_\Gamma)(\mu_X/\mu_\Gamma)^2 + (n_L/n_\Gamma)(\mu_L/\mu_\Gamma)^2}{1 + (n_X \mu_X)/(n_\Gamma \mu_\Gamma) + (n_L \mu_L)/(n_\Gamma \mu_\Gamma)} \quad (5.18)$$

Assuming that the Boltzmann statistics are valid for the electron concentrations in the crystals studied, the following approximation is also valid.

$$n_{\Gamma} = N_c^T \exp(E_F/K_B T) \quad (5.19)$$

and

$$\frac{n_L}{n_{\Gamma}} = \left(\frac{m_L^d}{m_{\Gamma}^d}\right)^{3/2} \exp(-\Delta E_{\Gamma L}/K_B T) \quad (5.20)$$

and

$$\frac{n_X}{n_{\Gamma}} = \left(\frac{m_X^d}{m_{\Gamma}^d}\right)^{3/2} \exp(-\Delta E_{\Gamma X}/K_B T) \quad (5.21)$$

where $\Delta E_{\Gamma X}$ and $\Delta E_{\Gamma L}$ are the Γ -X and Γ -L intervalley separation and m^d is the density of state effective mass. The empirical formulae for the conduction and density of state effective masses are as follows [88]:

$$m_{\Gamma}^d = 0.067 + 0.083x \quad (5.22)$$

$$m_X^d = 0.85 - 0.14x \quad (5.23)$$

$$m_L^d = 0.56 + 0.1x \quad (5.24)$$

The energy band gaps of the three different conduction band minima are given respectively by

$$E_g^{\Gamma} = \begin{cases} 1.424 + 1.247x & \text{for } x \leq 0.45 \\ 1.424 + 1.247x + 1.147(x - 0.45)^2 & \text{for } 0.45 < x \end{cases}$$

$$E_g^X = 1.900 + 0.125x + 0.143x^2 \quad (5.25)$$

$$E_g^L = 1.708 + 0.642x \quad (5.26)$$

Figures 5.1 and 5.2 show the results of electron and hole mobilities respectively for the Al composition between 0.10 and 0.45 and three different N_sQ values. From Table 5.1, it is clear that our results are in better agreement with the experimental data than those of others. The major discrepancy among these data is the calculation of the space charge scattering. The product of the density of the scattering center (N_s) and the capture cross section (σ) used in our calculations is $4.5 \times 10^4 \text{ cm}^{-1}$ which is less than those given in references [81] and [83].

Figures 5.3 and 5.4 show the room temperature electron and hole mobilities of different scattering processes as a function of Al compositions at $N_D = 1.5 \times 10^{16}$ and $N_A = 10^{18} \text{ cm}^{-3}$ respectively. According to Fig. 5.3, polar optical phonon scattering, intervalley scattering and space charge scattering are the three dominant scattering processes limiting the electron mobility at room temperature. However, polar optical phonon scattering, alloy scattering and deformation potential scattering are the three dominant scattering processes for hole mobility as shown in Fig. 5.4.

The Al composition and temperature dependence of the electron and hole mobilities of $\text{Al}_x\text{Ga}_{1-x}\text{As}$ at $N_D = 10^{17}$ and $N_A = 10^{18} \text{ cm}^{-3}$ are shown in Fig. 5.5 and Fig. 5.6 respectively. According to our calculations, ionized impurity scattering and space charge scattering are the two dominant processes at low temperatures 100K and 200K, for the electron mobility case. As the temperature increases, polar optical phonon scattering and intervalley scattering become more dominant. For the hole mobility case, deformation scattering and ionized impurity scattering and alloy

scattering are the three dominant processes at low temperature. When the temperature increases, the polar optical phonon scattering takes over the role of the ionized impurity scattering in limiting the hole mobility of $\text{Al}_x\text{Ga}_{1-x}\text{As}$.

Figures 5.7 and 5.8 show the temperature and doping density dependence of the electron and hole mobilities of $\text{Al}_{0.38}\text{Ga}_{0.62}\text{As}$. It is indicated that the doping level has little effect on either electron or hole mobilities of $\text{Al}_{0.38}\text{Ga}_{0.62}\text{As}$ if doping density is less than 10^{18} cm^{-3} . For doping density above 10^{18} cm^{-3} , however, there is a huge decrease. This is because at low temperature and high doping level, the ionized impurity scattering is the most dominant process among all scattering processes.

5.4 Summary

This chapter presents the results of a comprehensive study of the scattering processes of $\text{Al}_x\text{Ga}_{1-x}\text{As}$ alloy system. Calculations of the electron and hole mobilities of $\text{Al}_x\text{Ga}_{1-x}\text{As}$ as a function of doping density, temperature and Al composition have been carried out. It has been found that polar optical phonon scattering, intervalley scattering and space charge scattering are the three dominant processes for electron mobility of $\text{Al}_x\text{Ga}_{1-x}\text{As}$ at room temperature. As for the low temperature and high doping density such as 10^{18} cm^{-3} or higher, ionized impurity becomes dominant. For the hole mobility case, it was found that the influences of the alloy scattering, polar optical phonon scattering and deformation scattering are significant. Our theoretical calculations show good agreements with the experimental data.

Table 5.1 Electron mobilities of the $\text{Al}_{0.38}\text{Ga}_{0.62}\text{As}$ and $\text{Al}_{0.19}\text{Ga}_{0.81}\text{As}$.

x(%)	$N_D(\text{cm}^{-3})$	$\mu (\text{cm}^2/\text{V-s})$	$N_s Q(\text{cm}^{-1})$	experiment
0.38	1.5×10^{16}	1432	45000	1200
0.38	1.5×10^{16}	855	244000[78]	-
0.38	1.5×10^{16}	1231	95000 [80]	-
0.19	1.5×10^{16}	2689	45000	2700
0.19	1.5×10^{16}	2221	95000	-
0.19	1.5×10^{16}	1989	128500	-

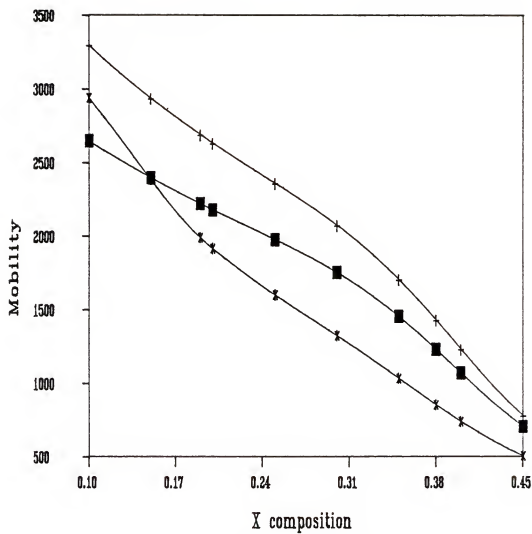


Fig. 5.1 Electron mobilities vs. Al composition and NsQ values. + : this study, solid box : NsQ obtained from [81] and * : NsQ obtained from [83].

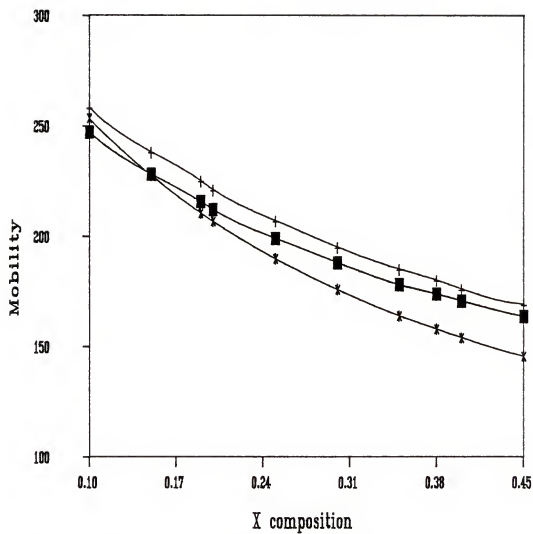


Fig. 5.2 Hole mobilities vs. Al composition and NsQ values. + : this study, solid box : NsQ obtained from [81] and * : NsQ obtained from [83].

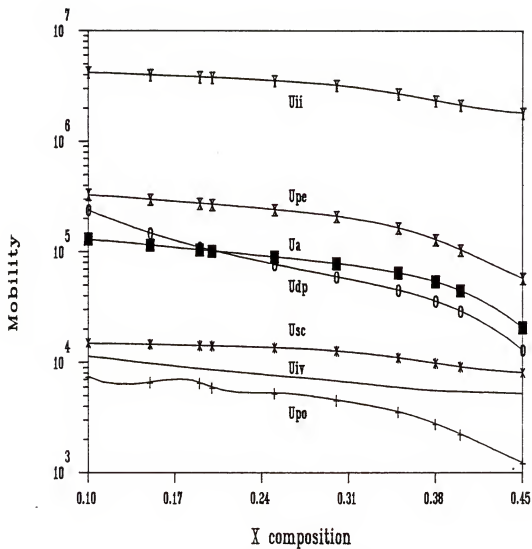


Fig. 5.3 Room temperature electron mobilities of different scattering processes vs. Al composition at $N_D = 1.5 \times 10^{16} \text{ cm}^{-3}$.

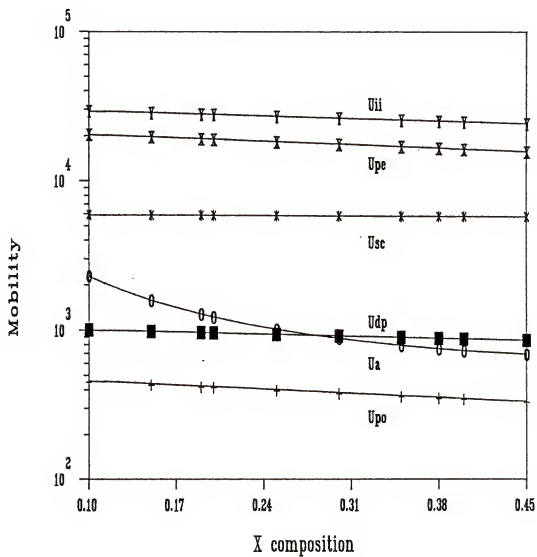


Fig. 5.4 Room temperature hole mobilities of different scattering processes vs. Al composition at $N_A = 10^{18} \text{ cm}^{-3}$.

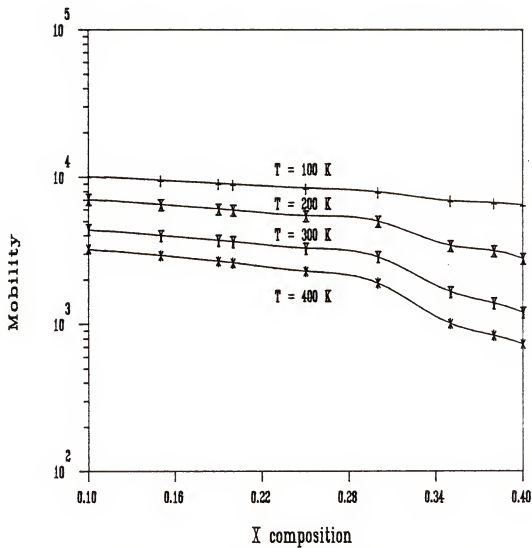


Fig. 5.5 Al composition and temperature dependence of electron mobility of $\text{Al}_x\text{Ga}_{1-x}\text{As}$ at $N_D = 10^{17} \text{ cm}^{-3}$.

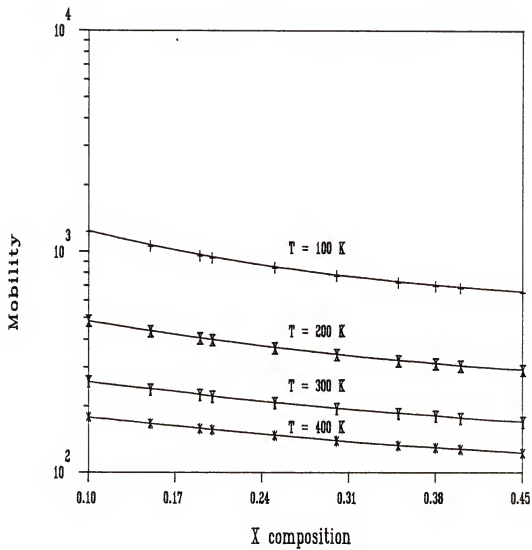


Fig. 5.6 Al composition and temperature dependence of hole mobility of $\text{Al}_x\text{Ga}_{1-x}\text{As}$ at $N_A = 10^{18} \text{ cm}^{-3}$.

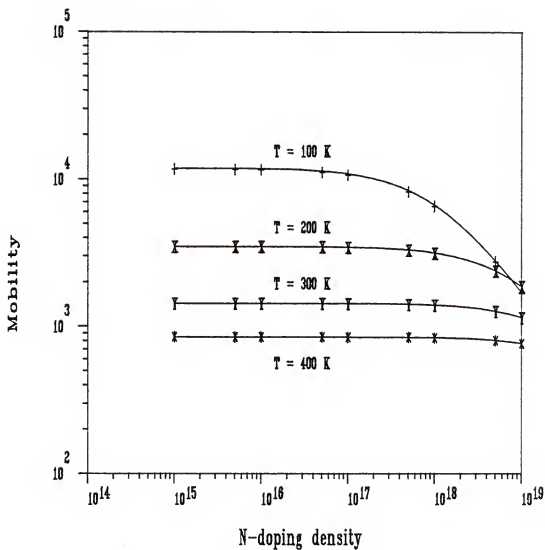


Fig. 5.7 Temperature and doping density dependence of electron mobility of $\text{Al}_{0.38}\text{Ga}_{0.62}\text{As}$.

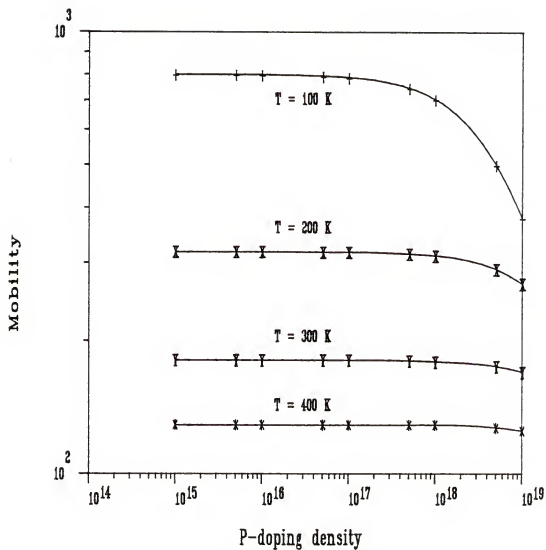


Fig. 5.8 Temperature and doping density dependence of hole mobility of $\text{Al}_{0.38}\text{Ga}_{0.62}\text{As}$.

CHAPTER 6 SUMMARY, CONCLUSION AND RECOMMENDATIONS

6.1 Summary and Conclusion

In this dissertation, an expert system has been developed to facilitate the optimal design of single-junction and multijunction solar cells for both space and terrestrial applications. A rule-base expert system was set up by adopting the experimental data and/or semiempirical formulae used today. In addition, a modified Box complex algorithm with some heuristic rules was implemented to reduce the computation time of the optimization process. This work was necessitated by the lack of an efficient optimization algorithm and by the failure of an inadequate device model to realistically predict the optimal performance of the solar cells.

A simple model for calculating the displacement damage of the proton or electron irradiated solar cells under different fluences, energies and environmental conditions has been presented in Chapter 2. A computer program for computing the degradation of short-circuit current, open-circuit voltage and conversion efficiency of a solar cell has been coded. This program included the calculations of the damage constant of the minority carrier diffusion length, a parameter important for the optimal design of irradiated solar cells. Close agreement was obtained between the calculated and measured I_{sc} , V_{oc} and η_c in the proton and electron irradiated $Al_{0.33}Ga_{0.67}As$ and $GaAs$ single-junction and two-junction cells for proton energies of 100 KeV up to 10 MeV and fluences varying from 10^{10} to 10^{12} cm^{-2} , and for 1 MeV electrons with fluences of 10^{14} , 10^{15} and 10^{16} cm^{-2} for both the normal and omnidirectional incident conditions.

In Chapter 3, a new computer model for truly optimizing the structure of $GaAs$ single-junction solar cell for both space and terrestrial applications is proposed. The

model, however, can apply easily to other solar cell system as well. This model not only takes into account the effects of the intrinsic structure parameters such as junction depth X_j , cell thickness T_j , doping densities N_A and N_D , surface recombination velocity S_p and S_n , but also incorporates the extrinsic structure parameters. An efficient Box optimization algorithm [15] with minor modifications is implemented in the model. The optimal efficiency of the single-junction GaAs solar cell obtained by the simulation is 27.8% at room temperature for AM0 which is close to the cell efficiency namely 26% made by the state of the art [69].

In Chapter 4, an expert system approach to the optimal design of multijunction solar cells for both terrestrial and space applications is described for the first time. An expert systems is a knowledge-intensive computer program. The knowledge of an expert system consists of facts and rules. The facts constitute a body of information that is widely shared, publicly available and generally agreed upon by experts in the field [33, 34, 68]. The rules are those if-then rules that characterize expert level decision making in the field. We have developed a rule-base system and a couple of heuristic rules which can prune the search space of the design problem and hence to save the computation time. The optimal designs of $Al_{0.44}Ga_{0.56}As/GaAs$ two-junction solar cell and $Al_{0.44}Ga_{0.56}As/GaAs/In_{0.53}Ga_{0.47}As$ three-junction solar cell with room temperature efficiencies for AM0 of 30.01% and 35.3% respectively were obtained in the simulation.

In Chapter 5, a comprehensive study of the scattering processes of $(AlGa)As$ has been presented. Theoretical calculations of the electron and hole mobilities in $Al_xGa_{1-x}As$ were made. It was found that the mobilities of $Al_xGa_{1-x}As$ are quite sensitive to the Al composition, temperature, doping density and defect density. This knowledge is indispensable to the fabrication of a high quality $(AlGa)As$ material and a high efficiency top cell of the GaAs based multijunction solar cells.

6.2 Recommendations

Although an expert system has been developed in this dissertation to accomplish the goals of optimizing the performance of single-junction and multijunction solar cells, there are some uncertainties due to the lack of experimental data. Actual cells should be fabricated and characterized in order to facilitate comparisons with the simulations results. Further extensions of this research include the following:

1. Since the threshold energy T_d plays a major role in the displacement damage model, it is important that an accurate value of T_d be obtained for each material used in a multijunction solar cell. Except for GaAs and Ge, values of T_d for materials used in the present model are still not well known. Also new data are needed to obtain more accurate calculations of the displacement damages in multijunction solar cells.
2. The path length and penetration depth are based on Janni's data [41]. Further experimental data for (AlGa)As and (InGa)As are needed to improve the computer simulations.
3. Accurate recombination cross section data for (AlGa)As, (InGa)As and Ge are needed for calculations of short-circuit degradation. This can be achieved by using DLTS technique to determine the recombination cross section in each cell.
4. The correlations between the minority carrier lifetime, diffusion length and mobility and doping density, temperature and x composition for $\text{In}_x\text{Ga}_{1-x}\text{As}$ and $\text{Al}_x\text{Ga}_{1-x}\text{As}$ should be further improved in order to optimize the top and bottom cells.
5. Additional device structures such as heterojunctions or three-terminal structures etc. should be considered for the expert system in the future.

6. Defect characterization of (InGa)As and (AlGa-)As should be done in order to extract the parameters which affect the quality of these materials.

In short, the more facts and rules an expert system has, the more powerful it is. The expert system presented in this dissertation may be considered simply as a kernel. As the information listed just above becomes available , the more accurate results can be derived from the expert system developed by the research described in this dissertation.

APPENDIX A A COMPUTER PROGRAM FOR CALCULATING THE TOTAL NUMBER OF DISPLACEMENT DEFECTS

```

{
The purpose of this program is to calculate the total number of
displacement defects induced by energetic electron or proton. The
input parameters are atomic number, atomic weight and threshold
energy of the cell material. The range of proton energy is from
400 eV to 10 MeV and is from 200 Kev to 5 MeV for electron.
}

Program Displace(input,output,out(lfn=103),inpa(lfn=104) );
CONST
    Mp   = 1.67264E-27;      (* proton mass, Kg *)
    Me   = 9.1095E-31;      (* electron mass,Kg *)
    Er   = 13.6;             (* Rydberg constant, eV *)
    Z1   = 1;                (* projectile atomic number *)
    a0   = 5.2917E-5 ;      (* Bohr's radius,um *)
    pai  = 3.141592654;
    Ne   = 4.42E10;          (* electron concentration,1/(um*um) *)
    Vc   = 2.99792E8;        (* velocity of light, m *)

TYPE
    Xout = array[1..50] of real;
    Yout = array[1..50] of real;
    projectile=(proton,electron);
    InPara = array [1..4] of real;

VAR
    out, inpa : text;        (* files to store the results *)
    Tm,        (* maximun transfer energy,eV *)
    Alpha,     (* Z2 / 137 *)
    Beta ,     (* velocity / (Vc of light) *)
    Beta2,     (* Beta * Beta *)
    Vd ,       (* average number of displacement *)
    TmDivTd,   (* Tm / Td *)
    PaiAB,     (* pai * alpha * beta *)
    Ein       : real ;      (* particle energy,eV *)
    Particle  : projectile; (* proton or electron *)
    CrossS    ,            (* crosssection area *)

```

```

Dtotal      ,      (* total displacements due to particle damage *)
Tbar        ,      (* average recoil energy *)
Ed          ,      (* displacement with no multiple scattering *)
ReduceC     ,
Dave        : xout;  (* displacement with multiple scattering *)
Ec          : real;  (* classical energy of  $M0*Vc**2$ , eV *)
ii          : integer;
JoeV        : real;  (* energy conversion, from joules to eV *)
Delta       : real;
cell,
FigNum      : integer;
keep        : xout;
Td          ,      (* Threshold energy *)
Z2          ,      (* Projectile atomic number *)
M2          : real ;  (* Projectile atomic weight *)
Sequence    : integer;
Range       : integer;

procedure initial;
var
  i, j : integer;
  ch,
  ans : char;
begin
  alpha:= Z2 / 137;
  Ec:= Me * sqr(Vc) / ( 1.60218E-19 );
  JoeV:= 1 / ( 1.60218E-19);
  Delta:= 4*pai* sqr(a0*Er*Z2);
end;

Function BetaRatio( Evar : real ): real;
var
  square : real;
begin
  square:= sqr( Evar/ Ec + 1);
  BetaRatio:=sqrt(1 - 1/square);
end;

PROCEDURE PreCal(index: projectile);
begin
  case index of
    proton : Tm:=( 4*M2) * Ein / Sqr(1+M2);
  electron: begin
    Tm:=2*Ein/(Mp*m2*sqr(Vc)*Joev)*(Ein+2*Me*sqr(vc)*Joev);

```

```

        beta:=Betaratio(Ein);
        beta2:=sqr(beta);
        TmDivTd:=Tm / Td;
        PaiAB:=Pai*alpha*beta;
    end;
end; (* case *)

end;

(* Calculation of the displacement cross sections *)

function CrossSection(index: projectile):real;
var
    Ta, Tb : real;
begin
    case index of
        proton : begin
            PreCal(index);
            if Td>=Tm then CrossS[ii]:=1E-16   else
                CrossS[ii]:= delta/(M2*Ein)* ( 1/Td - 1/Tm);
            CrossSection:=CrossS[ii];
        end;
        electron : begin
            PreCal(index);
            crossS[ii]:= (delta/sqr(Ec)) * (1-beta2)/ sqr(beta2) *
                (TmDivTd -1 -beta2*ln(TmDivTd)+ 2*PaiAB*
                (sqrt(TmDivTd)-1)- PaiAB*ln(TmDivTd) );
            CrossSection:=CrossS[ii];
        end;
    end;
end; (* of case *)

end;

procedure RecoilAve(index:projectile);
var
    Numer, Denom : real;
begin
    if index=electron then
        begin
            Numer:= Tm*ln(TmDivTd)- beta2*(Tm-Td)+ 2*PaiAB*(Tm-sqrt(Tm*Td))
                - PaiAB*(Tm-Td);
            Denom:= TmDivTd- 1- Beta2*ln(TmDivTd)+ 2*PaiAB*(sqrt(TmDivTd)-1)
                - PaiAB*ln(TmDivTd);
            Tbar[ii]:= Numer / Denom;
        end
    else
        Tbar[ii]:= (Td*Tm)/(Tm-Td) * ln(tm/td);
    end;
end;

```

(* First derivative of the path length. *)

```

function Dp(Evar:real):real;
var
  xx : real;
begin
  xx:=Evar / 1E6;
  if cell=1 then
  begin
    if XX <= 0.150 Then
      Dp:=1.938958*EXP(-0.452449*LN(XX))    else
    if xx <= 1.25  then
      Dp:=12.74343*EXP(0.145135*LN(XX))    else
      Dp:=15.95597*EXP(0.5556635*LN(XX));
    end else
    if cell = -2  then
    begin
      if xx <= 0.150  then
        Dp:= 1.940185*exp(-0.4521705*ln(xx))  else
      if xx <= 1.25  then
        Dp:= 12.819498*exp(0.1469979*ln(xx))  else
        Dp:= 16.035977*exp(0.5565806*ln(xx));
      end else
      if cell=2 then
      begin
        IF XX <= 0.150 THEN
          DP:=2.106838*EXP(-0.454091*LN(XX))    ELSE
        IF XX <= 1.25  THEN
          DP:=13.455826*EXP(0.135261*LN(XX))    ELSE
          DP:=16.933599*EXP(0.550638*LN(XX));
        end else
        if cell=3 then
        begin
          if xx <= 0.150 then
            Dp:=1.9473111*EXP(-0.453661*LN(XX))  else
          if xx <= 1.25  then
            Dp:=12.496518*EXP(0.137668*LN(XX))  else
            Dp:=15.817996*EXP(0.545829*LN(XX));
          end else
          begin
            if xx <= 0.150 then
              Dp:=2.0464761*EXP(-0.452753*LN(XX))    else
            if xx <= 1.25  then

```

```

        Dp:=13.265437*EXP(0.141380*LN(XX))      else
        Dp:=1.6697369*EXP(0.562229*LN(XX));
    end;

end;

(* first derivative of penetration depth *)

function DrE(Evar: real):real;
var xx : real;
begin
    xx := Evar / 1.0E6;
    if cell = -2 then
    begin
        if xx <= 0.20 then
            Dre:= 104.1110 + 7388.5995 * xx - 15681.3270 *sqr(xx)  else
            if xx <= 1.0 then
                Dre:= 734.929 + 1661.5505 * xx - 1069.1373 * sqr(xx)  else
                Dre:= 1463.058 - 81.84514 * xx - 2.55829 * sqr(xx);
            end else
            if cell = -1 then
            begin
                if xx <= 0.20 then
                    Dre:= 103.2968 + 7322.016 * xx - 15542.8868 * sqr(xx)  else
                    if xx <= 1.0 then
                        Dre:= 728.088 + 1647.176 * xx - 1060.1532 * sqr(xx)  else
                        Dre:= 1449.811 - 81.27775 * xx + 2.540295 * sqr(xx);
                    end;
                if cell = 0 then
                begin
                    if xx <= 0.20 then
                        Dre:= 101.9048 + 7298.573 * xx - 15467.2737 * sqr(xx)  else
                        if xx <= 1.0 then
                            Dre:= 727.348 + 1638.159 * xx - 1051.4330 * sqr(xx)  else
                            Dre:= 1446.344 - 79.51220 * xx + 2.484936 * sqr(xx);
                        end;
                    if cell = 1 then
                    begin
                        if xx <= 0.20 then
                            Dre:= 103.5243 + 7340.605 * xx - 15581.5448 * sqr(xx)  else
                            if xx <= 1.0 then
                                Dre:= 707.2341 + 1755.050 * xx - 1166.50767* sqr(xx)  else
                                Dre:= 1453.507 - 81.43596 * xx + 2.5452710* sqr(xx);
                            end;
                        if cell = 2 then

```



```

begin
  if xx <= 0.20 then
    Dre:= 100.0425 + 7057.603 * xx - 1499.25345 * sqr(xx) else
    if xx <= 1.0 then
      Dre:= 700.994 + 1589.943 * xx - 1024.50218* sqr(xx) else
      Dre:= 1397.202 - 78.99068 * xx + 2.4680365* sqr(xx);
end;
if cell = 3 then
begin
  if xx <= 0.20 then
    Dre:= 99.67980 + 6.757367 * xx - 14485.9866 * sqr(xx) else
    if xx <= 1.0 then
      Dre:= 678.5045 + 1476.745 * xx - 963.79863 * sqr(xx) else
      Dre:= 1321.854 - 83.65355 * xx + 2.756492 * sqr(xx);
end;
if cell = 4 then
begin
  if xx <= 0.20 then
    Dre:= 109.6292 + 7729.241 * xx - 16443.7531 * sqr(xx) else
    if xx <= 1.0 then
      Dre:= 769.090 + 1729.35898*xx - 1118.0255 * sqr(xx) else
      Dre:= 1522.400 - 86.90735 * xx + 2.7739247* sqr(xx);
end;
end;

function VdE( index:projectile): real;
begin
  case index of
    proton: begin
      Precal(index);
      if Tm > ( 2*Td ) then
        VDE:=1/2*(TM/(TM-TD))* ( 1+ LN(TM/(2*TD))) ELSE
        if Tm > Td then
          VdE:=1.0 else
          VdE:=0.0;
      end;
    electron:begin
      particle:=electron;
      Precal(index); ii:=1; RecoilAve(particle);
      if Tbar[1] > ( 2 * Td ) then
        VdE:= Tbar[1] / (2*Td) else
        if Tbar[1] >= Td then
          VdE:=1.0 else
          VdE:=0.0;
    end;
  end;
end;

```

```

        end;
        end; (* case *)
end;

(* Calculate total number of displacement defect *)

function TotalD(index:projectile):real;
var
    x0, x1, x2, xi, h : real;
    n, jj : integer;
    x00 : real;
begin
    n:=250;
    case index of
        proton : begin
            h:=Ein/ ( 2 * n );
            if Ein< 250 then x0:=0.0 else
            x0:=crosssection(index)*VdE(index)*Dp(Ein);
            x1:=0.0; x2:=0.0;
            for jj:=1 to ( 2*n -1 ) do
                begin
                    Ein:=0.0 + jj*h;
                    if Ein< 200 then begin end else
                    if odd(jj) then
                        x1:=x1+crosssection(index)*VdE(index)*Dp(Ein) else
                        x2:=x2+crosssection(index)*VdE(index)*Dp(Ein);
                    end;
                    TotalD:= 1E-6 * Ne*h*(x0+2*x2+4*x1)/3;
                end;
            end;
        electron: begin
            h:=Ein / ( 2 * n );
            if Ein< 260E3 then begin x0:=0.0 ; end else
            x0:=crosssection(index)*Vde(index)*Dre(Ein);
            if x0<=0.0 then x0:=0.0;
            x1:=0.0; x2:=0.0;
            for jj:=1 to (2*n - 1) do
                begin
                    Ein:=0.0 + jj*h;
                    if Ein< 260E3 then
                        begin end else
                    if odd(jj) then
                        x1:=x1+crosssection(index)*vde(index)*Dre(Ein) else
                        x2:=x2+crosssection(index)*Vde(index)*Dre(ein);
                    end;
                end;
            end;
        end;
    end;
end;

```

```

        ToTalD:= 1.0E-6*Ne*H*(x0+2*x2+4*x1)/3;
    end;
    end; (* case *)
end;

Procedure SetUpModel;
var i, j : integer;
    Rtemp: xout;
begin
    writeln('----> Part one. ');
    writeln(' -----> Td ? '); readln(Td);
    writeln(' -----> Z2 ? '); readln(Z2);
    writeln(' -----> M2 ? '); readln(M2);
    writeln(' -----> cell ?');readln(cell);
    initial;
    writeln(' 1 = Calculate proton displacement cross section');
    writeln(' 2 = Calculate electron displacement cross section');
    writeln(' 3 = average transferred energy for proton ');
    writeln(' 4 = average transferred energy for electron');
    writeln(' 5 = total number of displacement defect for proton ');
    writeln(' 6 = total number of displacement defect for electron');
    read (FigNum);
    case figNum of
    1 : begin
        for j:=1 to 50 do
            begin
                ii:=j;
                Ein:=exp((j/10 +2)*ln(10) );
                keep[j]:=j*0.1;
                IF Ein<=180 then
                    Ein:=180;
                Rtemp[j]:=( ln(CrossSection(proton)*1E16)) / LN(10);
            end;
        end;
    2 : begin
        for j:=1 to 50 do
            begin
                ii:=j;
                Ein:=200E3*j;
                keep[j]:=j*0.1;
                if Ein<=250 then
                    Ein:=250;
                if (ein<=360) and (cell=4) then
                    Ein:=360;
            end;
        end;
    end;
end;

```

```

        Rtemp[j]:=CrossSection(electron)*1E16;
    end;
end;
3 : begin
    PARTICLE:=proton;
    for J:=1 to 50 do
        begin
            ii:=j;
            Ein:=100E3 * j ;
            PreCal(proton);
            RecoilAve(proton);
            keep[j]:=Ein /1E6;
            Rtemp[j]:=Tbar[j];
        end;
    end;
4 : begin
    particle:=electron;
    for j:=1 to 50 do
        begin
            ii:=j;
            Ein:=100E3 * j ;
            PreCal(electron);
            RecoilAve(electron);
            keep[j]:=Ein/1E6;
            Rtemp[j]:=Tbar[j];
        end;
    end;
5 : begin
    writeln('..... range. 1--> 20 Kev.  2-->1 Mev.  3-->10 Mev ? ');
    readln(dnum);
    for j:=1 to 50 do
        begin
            ii:=j;
            if dnum=1 then
                Ein:=400*j      else
            if dnum=2 then
                Ein:=20000*j    else
            Ein:=200E3*j;
            Dtotal[j]:=TotalD(proton);
            keep[j]:=j / 10;
            Rtemp[j]:=Dtotal[j];
        end;
    end;
6 : begin

```

```

    for j:=1 to 50 do
    begin
        Ii:=J;
        Ein:=100E3*j;
        Dtotal[j]:=ToTalD(electron);
        keep[j]:=j/10;
        Rtemp[j]:=Dtotal[j];
    end;
end;
end; (* of case *)
for i:=1 to 50 do
    writeln(out,keep[i],rtemp[i]);
end;

begin (* of main *)
    writeln(' --> Start to calculation the displacement defect!');
    SetUpModel;
end.

```

APPENDIX B A COMPUTER PROGRAM FOR CALCULATING THE DEGRADATION OF SHORT-CIRCUIT CURRENT

```
{
This program is to calculate the short-circuit degradation
of single-junction and multijunction solar cells. In addition, the
calculation for damage constants of minority carrier diffusion
length and lifetime is included in this program. GaAs, Si, Ge,
InGaAs and AlGaAs are those materials used in this program.
Proton and electron are the energetical particles that produce
the displacement defects. This program is user friendly. User
may just follow the questions given by the program to enter the
necessary information for the program to execute.
}
```

```
Program SolarIrr(input,output,out(lfn=103),inpa(lfn=104),Inpb(lfn=105),
                Inp2(lfn=106),inp3(lfn=107),inp4(lfn=108));
```

```
CONST
```

```
  Mp   = 1.67264E-27;      (* proton mass, Kg *)
  Me   = 9.1095E-31;      (* electron mass,Kg *)
  Er   = 13.6;            (* Rydberg constant, eV *)
  Z1   = 1;               (* projectile atomic number *)
  a0   = 5.2917E-5 ;      (* Bohr's radius,um *)
  pai  = 3.141592654;
  Ne   = 4.42E10;         (* electron concentration,1/(um*um) *)
  Vc   = 2.99792E8;       (* velocity of light, m *)
  ln10 = 2.30258509;
```

```
TYPE
```

```
  Xout = array[0..50] of real;
  Yout = array[0..50] of real;
  projectile=(proton,electron);
  InPara = array [1..4] of real;
```

```
VAR
```

```
  out, inpa,                (* files to store the results *)
  Inp2, Inp3, Inp4,
  inpb   : text;
  Ein    : real ;          (* particle energy,eV *)
```

```

Particle : projectile;      (* proton or electron *)
ReduceC  ,
Kv3, Wl3,
kv2, Wl2,
Wl1, Kv1 : Xout;
Delta    : real;
ii       : integer;
Config,
Cell     : integer;
Plen     : real;
Rlen     : real;
Device   : integer;
Sequence : integer;
Range    : integer;
Wd,
Tj, Xj,
Lp, Lnn, LpIrr, LnIrr,
Nintr,
NA, ND,
Dpp, Dn,
TauP, TauN, TauNIrr, TauPIrr,
MrestN, MrestP,
Mneff, Mpeff,
Vther1, Vther2,
VthP, Vthn,
Klp, Kln,
Ktp, Ktn,
CapCn,
CapCp,
low,
high     : InPara;
tcell,
lower,
flux,
absorpt,
Dxij,
SigMaR  : real;
SigmaH,
SigmaE  : InPara;
Name    : array [1..3] of integer;
Omiter,
Aiter,
DIST    : INTEGER;
UpPene  : real;

```

```

Einn      : real;
Window    : array [1..3] of boolean;
Average,
BasicC,
OMNI,
Runn,
normal    : boolean;
absp      : array[1..200] of real;
IscRatio,
VscRatio,
Pmratio   : real;

procedure initial;
var
  i, j : integer;
  ch,
  ans  : char;
begin
  WRITELN(' Omni directional irradiation ? "Y" or "y"');
  readln(ch);
  if (Ch='y') or (Ch='Y') then
  begin
    Omni:=true ;
    writeln(' what is the omni iteration #, <= 20 ');
    readln(omiter);
    writeln(out,omiter);
  end   else
    Omni:=false;
  writeln(' cell configuration? 1, 2 or 3 ?');
  readln(Config);
  for i:= 1 to Config do
  begin
    writeln(' window layer ? "Y" or "y" ');
    readln(ch);
    if (ch='y') or (ch='Y') then
      window[i]:= true      else
      window[i]:= false;
    writeln( i:1, ' window layer thickness Wd ? ');
    readln(Wd[i]); writeln(out,' Wd [',i:1,'] = ', Wd[i]);
    writeln(i:1,' cell name: -2,-1,0,1,2,3,4,5 ');
    writeln(' -2 : AlGaAs: Al =40% , -1 : 33%, 0 : 85%, 1: 35% ');
    writeln(' 2 : GaAs, 3: InGaAs , 4 ; Ge, 5: Si ');
    readln(Name[i]);
    writeln( i:1,' Junction depth Xij ? ');

```



```

    readln (Xj[i]); writeln(out,' Xj [' ,i:1,'] = ', Xj[i]);
    writeln( i:1,' Cell thickness Tj ? ');
    readln (Tj[i]); writeln(out,' Tj [' ,i:1,'] = ', Tj[i]);
    writeln( i: 1, ' low and high cutoff wavelength ');
    readln( low[i], high[i]);
    writeln( 1: 1, ' Electron and Hole capture cross sections? ');
    readln ( SigmaE[i],SigmaH[i]);
    writeln(out,' .. SigmaE = ',SigmaE[i], ' .. SigmaH = ',SigmaH[i]);
end;
for i:= 0 to 23 do
    readln(inp1,WL1[i],Kv1[i]); (* for GaAs *)
for i:= 0 to 35 do
    readln(inp2,WL2[i],Kv2[i]); (* for Si *)
for i:= 0 to 11 do
    readln(inp3,WL3[i],Kv3[i]); (* for Ge *)
{
for i:= 1 to 182 do
    read(inp4,Atype[i],Btype[i]);
}
end;

(* GaAs absorption coefficients vs. wavelength *)

Function GaAsAbsorp( Atemp : real ): real;
var
    i : integer;
    Btemp : real;
    found : boolean;
begin
    if Atemp <= 0.300 then Atemp:=0.300;
    if Atemp >= 0.873 then Atemp:=0.873;
    found:=false;
    i:= 1;
    repeat
        if Atemp <= WL1[i] then
            begin
                Btemp:= Kv1[i-1]-(Kv1[i-1]-kv1[i])/(WL1[i]-WL1[i-1])
                    *(Atemp-WL1[i-1]);
                found:= true;
            end
            else
                i:= i + 1 ;
        until ( i >= 24 ) or ( found );
    GaAsAbsorp:= 4.0 * 3.141596 * Btemp / Atemp ;
end;

```

(* Si absorption coefficients vs. wavelength *)

```
Function SiAbsorp( Atemp : real ): real;
var
  i : integer;
  Btemp : real;
  found : boolean;
begin
  if Atemp <= 0.294 then Atemp:=0.294;
  if Atemp >= 1.125 then Atemp:=1.125;
  found:=false;
  i:= 1;
  repeat
    if Atemp <= WL2[i] then
      begin
        Btemp:= Kv2[i-1]-(Kv2[i-1]-kv2[i])/(WL2[i]-WL2[i-1])
          *(Atemp-WL2[i-1]);
        found:= true;
      end
      else
        i:= i + 1 ;
    until ( i >= 36 ) or ( found );
    SiAbsorp:= 4.0 * 3.141596 * Btemp / Atemp ;
  end;
```

(* Ge absorption coefficients vs. wavelength *)

```
Function GeAbsorp( Atemp : real ): real;
var
  i : integer;
  Btemp : real;
  found : boolean;
begin
  if Atemp <= 0.500 then Atemp:=0.500;
  found:=false;
  i:= 1;
  repeat
    if Atemp <= WL3[i] then
      begin
        Btemp:= Kv3[i-1]-(Kv3[i-1]-kv3[i])/(WL3[i]-WL3[i-1])
          *(Atemp-WL3[i-1]);
        found:= true;
      end
      else
        i:= i + 1 ;
```

```

until ( i >= 12 ) or ( found );
GeAbsorp:= 4.0 * 3.141596 * Btemp / Atemp ;
end;

(* (InGa)As absorption coefficients vs. wavelength *)
(* In --> 53 % ; Ga --> 47 % *)

Function InGaAsAbsorp(Atemp : real):real;
var
    Btemp : real;
begin
    if Atemp <= 0.74 then
        Btemp:= 10.0 else
    if Atemp <= 1.40 then
        Btemp:= exp((-1.51515*Atemp+6.12121)*ln10) / 1E4 else
    if Atemp <= 1.50 then
        Btemp:= exp((-1.5491*Atemp+6.16874)*ln10) / 1E4 else
    if Atemp <= 1.54 then
        Btemp:= exp((-3.65495*Atemp+9.32752)*ln10) / 1E4 else
    if Atemp <= 1.73 then
        Btemp:= exp((-14.2051*Atemp+25.5749)*ln10) / 1E4 else
        Btemp:= 1E-3;
    InGaAsAbsorp:= Btemp;
end;

(* Path length vs. proton energy *)

Function Plength(Evar : real ): real;
var
    xx : real;
begin
    xx:= Evar / 1E6;
    if cell=0 then
        begin
            if xx <= 0.150 then
                plength:= 3.300891*EXP(0.550212*LN(XX)) ELSE
            if xx <= 1.250 then
                plength:= 10.796238*EXP(1.163227*LN(XX)) ELSE
                plength:= 9.963561*EXP(1.565366*LN(XX));
            end else
            IF CELL=-2 THEN
                begin
                    if xx <= 0.150 then
                        plength:= 3.541585*exp(0.5478295*ln(xx)) else

```

```

    if xx <= 1.250 then
        Plength:= 11.176566*exp(1.1469979*ln(xx)) else
        Plength:= 10.302054*exp(1.5565806*ln(xx));
end else
if cell = 1 then
begin
    if xx <= 0.150 then
        Plength:= 3.541146*EXP(0.547551*LN(XX)) ELSE
    if xx <= 1.250 then
        Plength:= 11.12832*EXP(1.145135*LN(XX)) ELSE
        Plength:= 10.25669*EXP(1.555663*LN(XX));
end else
if cell = 2 then
begin
    if xx <= 0.150 Then
        Plength:=3.859321*EXP(0.545909*LN(XX)) ELSE
    if xx <= 1.25 THEN
        plength:=11.852628*EXP(1.135261*LN(XX)) ELSE
        plength:=10.920408*EXP(1.550638*LN(XX));
end else
if cell = 3 then
begin
    if xx <= 0.150 then
        Plength:= 3.564291*EXP(0.546339*LN(XX)) ELSE
    if xx <= 1.25 then
        Plength:= 10.98432*EXP(1.137668*LN(XX)) ELSE
        plength:= 10.23269*EXP(1.545829*LN(XX));
end else
begin
    if xx <= 0.150 then
        Plength:= 3.739584*EXP(0.547247*LN(XX)) ELSE
    if xx <= 1.25 then
        Plength:= 11.62228*EXP(1.141380*LN(XX)) ELSE
        Plength:= 10.68817*EXP(1.562229*LN(XX));
end;
end;

```

(* Penetration length vs. proton energy *)

Function Rlength(Evar: real) :real;

Var

xx : real;

begin

xx:=Evar / 1E6 ;

```

if cell=0 then
begin
  if xx <= 0.175 then
    rlength:= 5.010253*EXP(0.865712*LN(XX))  ELSE
  if xx <= 1.5  then
    rlength:= 10.310898*EXP(1.257302*LN(XX)) ELSE
    rlength:= 9.561796*EXP(1.579760*LN(XX));
end  else
if cell=-2 then
begin
  if xx <= 0.175  then
    rlength:= 5.378740*exp(0.873210*ln(xx))  else
  if xx <= 1.50  then
    rlength:= 10.632386*exp(1.2475845*ln(xx))  else
    rlength:= 9.8560927*exp(1.5722024*ln(xx));
end  else
if cell =1 then
begin
  if xx <= 0.175 then
    Rlength:= 5.378855*EXP(0.874058*LN(XX))  ELSE
  if xx <= 1.5  then
    Rlength:= 10.58202*EXP(1.24645*LN(XX))  ELSE
    Rlength:= 9.809475*EXP(1.57141*LN(XX));
end  else
if cell = 2 then
begin
  if xx <= 0.175  then
    Rlength:=5.86137*EXP(0.8786719*LN(XX))  ELSE
  if xx <= 1.5  then
    Rlength:=11.236521*EXP(1.243952*LN(XX))  ELSE
    Rlength:=10.427194*EXP(1.567030*LN(XX));
end else
if cell = 3 then
begin
  if XX <= 0.175  then
    Rlength:= 5.244473*EXP(0.877962*LN(XX))  ELSE
  if Xx <= 1.5  then
    Rlength:= 10.34555*EXP(1.249145*LN(XX))  ELSE
    Rlength:= 9.707031*EXP(1.563826*LN(XX));
end else
begin
  if xx <= 0.175  then
    Rlength:= 5.640120*EXP(0.871511*LN(XX))  ELSE
  if xx <= 1.5  then

```

```

      Rlength:= 11.05239*EXP(1.242816*LN(XX))   ELSE
      Rlength:= 10.22688*EXP(1.577814*LN(XX));
end;
end;

(* reduced energy vs. path length and penetration depth *)

Function eeleft(xx : real; kk : integer):real;
var dx : real;
    eleft: real;
begin
  if kk=1 then dx:=Rlen-xx else dx:=Plen-xx;
  if KK=1 then
    begin
      if Xx >= Rlen then eleft:=0.0 else
        if cell = 0 then
          begin
            if Ein <= 0.1E6 then
              eleft:= 0.159627*EXP(1.164662*LN(DX))   ELSE
            if Ein <= 0.45E6 then
              eleft:= 0.143353*EXP(0.892092*LN(DX))   ELSE
            IF EIN <= 2.0E6 THEN
              ELEFT:= 0.179735*EXP(0.722203*LN(DX))   ELSE
              eleft:= 0.256094*EXP(0.619708*LN(DX));
            end else
              if cell = -2 then
                begin
                  if Ein <= 0.1E6 then
                    eleft:= 0.1488971*exp(1.1532964*ln(dx)) else
                  if Ein <= 0.45E6 then
                    eleft:= 0.1371156*exp(0.9002795*ln(dx)) else
                  if Ein <= 2.0E6 then
                    eleft:= 0.17390829*exp(0.726625*ln(dx)) else
                    eleft:= 0.2422937*exp(0.6300121*ln(dx));
                end else
                  if cell = 1 then
                    begin
                      if Ein <= 0.1E6 then
                        Eleft:= 0.149104*EXP(1.152006*LN(DX))   ELSE
                      if Ein <= 0.45E6 then
                        Eleft:= 0.1374183*EXP(0.901323*LN(DX))   ELSE
                      IF EIN <= 2.0E6 THEN
                        ELEFT:= 0.174311*EXP(0.727081*LN(DX))   ELSE
                        Eleft:= 0.250527*EXP(0.622724*LN(DX));
                    end
                  end
                end
              end
            end
          end
        end
      end
    end
  end
end

```

```

end   else
if cell = 2 then
begin
    if Ein <= 0.1E6 then
        Eleft:= 0.136198*EXP(1.145118*LN(DX))      ELSE
    if Ein <= 0.45E6 then
        Eleft:= 0.1290153*EXP(0.907960*LN(DX))    ELSE
    IF EIN <= 2.0E6 THEN
        ELEFT:= 0.168008*EXP(0.724845*LN(DX))      ELSE
        Eleft:= 0.240366*EXP(0.6243284*LN(DX));
end   else
if cell = 3 then
begin
    if Ein <= 0.1E6 then
        Eleft:= 0.155403*EXP(1.148223*LN(DX))      ELSE
    if Ein <= 0.45E6 then
        Eleft:= 0.140691*EXP(0.901968*LN(DX))      ELSE
    IF EIN <= 2.0E6 THEN
        ELEFT:= 0.178959*EXP(0.722529*LN(DX))      ELSE
        Eleft:= 0.248245*EXP(0.627503*LN(DX));
end else
begin
    if Ein <= 0.1E6 then
        Eleft:= 0.140368*EXP(1.155406*LN(DX))      ELSE
    if Ein <= 0.45E6 then
        Eleft:= 0.131029*EXP(0.906509*LN(DX))      ELSE
    IF EIN <= 2.0E6 THEN
        ELEFT:= 0.168896*EXP(0.726630*LN(DX))      ELSE
        Eleft:= 0.245948*EXP(0.619878*LN(DX));
end;
end else
begin
    if Xx >= Plen then Eleft:=0.0 else
    if cell =0 then
    begin
        if Ein <= 0.07E6 then
            Eleft:= 0.1252061*EXP(1.875158*LN(DX))    ELSE
        if Ein <= 0.275E6 then
            Eleft:= 0.1023260*EXP(1.205425*LN(DX))    ELSE
        IF EIN <= 1.25E6 THEN
            ELEFT:= 0.142437*EXP(0.812657*LN(DX))      ELSE
            Eleft:= 0.230867*EXP(0.638220*LN(DX)) ;
        end 'else
    if cell = -2 then

```

```

begin
  if Ein <= 0.07E6 then
    Eleft:= 0.1080430*exp(1.8799742*ln(dx))      else
  if Ein <= 0.275E6 then
    Eleft:= 0.093611105*exp(1.2303236*ln(dx))    else
  if Ein <= 1.25E6 then
    Eleft:= 0.1352034*exp(0.82234935*ln(dx))      else
    Eleft:= 0.21969256*exp(0.64718419*ln(dx));
end else
if cell =1 then
begin
  if Ein <= 0.07E6 then
    Eleft:= 0.107870*EXP(1.880521*LN(DX))        ELSE
  if Ein <= 0.275E6 then
    Eleft:= 0.093579*EXP(1.233397*LN(DX))        ELSE
  IF EIN <= 1.25E6 THEN
    ELEFT:= 0.135318*EXP(0.823439*LN(DX))        ELSE
    Eleft:= 0.224582*EXP(0.642162*LN(DX));
end else
if cell = 2 then
begin
  if Ein <= 0.07E6 then
    Eleft:= 0.0907337*EXP(1.883710*LN(DX))        ELSE
  if Ein <= 0.275E6 then
    Eleft:= 0.0837655*EXP(1.251878*LN(DX))        ELSE
  IF EIN <= 1.25E6 THEN
    ELEFT:= 0.126622*EXP(0.829107*LN(DX))        ELSE
    Eleft:= 0.214667*EXP(0.644224*LN(DX));
end else
if cell = 3 then
begin
  if Ein <= 0.07E6 then
    Eleft:= 0.105743*EXP(1.883100*LN(DX))        ELSE
  if Ein <= 0.275E6 then
    Eleft:= 0.0925301*EXP(1.250373*LN(DX))        ELSE
  IF EIN <= 1.25E6 THEN
    ELEFT:= 0.135927*EXP(0.8255772*LN(DX))        ELSE
    Eleft:= 0.222678*EXP(0.646363*LN(DX));
end else
begin
  if Ein <= 0.07E6 then
    Eleft:= 0.097060*EXP(1.880410*LN(DX))        ELSE
  if Ein <= 0.275E6 then
    Eleft:= 0.0874221*EXP(1.2431922*LN(DX))        ELSE

```



```

        IF EIN <= 1.25E6 THEN
            ELEFT:= 0.1300987*EXP(0.8247081*LN(DX))
        ELSE
            Eleft:= 0.220143*EXP(0.6394407*LN(DX));
        end;
    end;
    if eleft<=0.0 then eeleft:=0.00 else eeleft:=eleft*1E6 ;
end;

(* total number of defects vs. proton energy *)

function Dcx(Evar:real):real;
var
    xx : real;
begin
    xx := Evar / 1E6;
    if cell=1 then
        begin
            if xx<= 0.00032 then
                Dcx:= 0.0      else
            if XX <= 0.004 then
                Dcx:=-3.7668 + 1.5337E4*xx -3.284372E6 *sqr(xx)
                    + 3.16018E8 * sqr(xx) *xx else
            if xx <= 0.2 then
                Dcx:=31.119 + 822.902*xx - 6.4357E3 * sqr(xx)
                    + 1.7539E4 * sqr(xx) * xx else
                Dcx:=77.465 + 19.003*xx - 0.8047*sqr(xx) + 0.033244*sqr(xx)*xx;
            end else
            if cell = -2 then
                begin
                    if xx <= 0.00032 then
                        Dcx:=0.0      else
                    if xx <= 0.004 then
                        Dcx:= -3.435537 + 2.8478580E4 * xx - 7.3789605E6 * sqr(xx)
                            + 7.26184506E8 *sqr(xx)*xx else
                    if xx <= 0.0136 then
                        Dcx:= 158.5301737*exp(0.24944024*ln(xx))      else
                    if xx <= 0.2 then
                        Dcx:= 96.805575 * exp(0.13378489*ln(xx))      else
                        Dcx:= 7.77309E1 + 6.1216539 * xx + 1.0583263 * sqr(xx)
                            - 0.06695623*sqr(xx)*xx;
                    end else
                if cell=2 then
                    begin
                        if xx <=0.00032 then

```

```

        Dcx:=0.0      else
    if xx <= 0.004 then
        Dcx:=-4.72641 + 4.0476E4*xx - 1.05160E7*sqr(xx)
            + 1.03643E9 *sqr(xx)*xx else
    if xx <= 0.0136 then
        Dcx:= 222.64114*exp(0.2473555*ln(xx)) else
    if xx <= 0.2 then
        Dcx:= 136.18292*exp(0.1320091*ln(xx)) else
        Dcx:=108.49 + 8.7396*xx + 1.2429*sqr(xx) - 0.078368*sqr(xx)*xx;
    end else
    if cell=3 then
    begin
        if xx<= 0.00032 then
            Dcx:=0.0      else
        if xx <= 0.004 then
            Dcx:=-5.63716 + 2.2414E4*xx -4.741112E6*sqr(xx)
                + 4.53575E8*sqr(xx)*xx      else
        if xx <= 0.2 then
            Dcx:=45.234 + 1.313E3 *xx - 1.0286E4* sqr(xx)
                + 2.7847E4*sqr(xx)*xx      else
            Dcx:=117.71 + 30.862*xx - 1.4795*sqr(xx) + 0.062118*sqr(xx)*xx;
        end else
        begin
            if xx<= 0.00096 then
                Dcx:=0.0      else
            if xx <= 0.004 then
                Dcx:=-1.3800 + 1.8865E3*xx + 7.39931E4*sqr(xx)
                    - 3.16944E7*sqr(xx)*xx else
            if xx <= 0.2 then
                Dcx:=7.4836 + 3.020E2*xx - 2.2730E3*sqr(xx)
                    + 5.5966E3*sqr(xx)*xx else
                Dcx:=25.334 + 10.060*xx - 0.6903*sqr(xx) + 0.03290*sqr(xx)*xx;
            end;
        end;
    end;

```

(* penetration length vs. electron energy *)

```

Function Erlen(E00:real):real;
var E0:real;
begin
    E0:= E00/1.0E6;
    if cell = -2 then (* 40% *)
    begin
        if E0 <= 0.2 then

```

```

Erlen:=-9.13188E-1 + 104.1110*E0 + 3694.2997*sqr(E0)
      - 5227.1090*sqr(E0)*E0  else
if E0 <= 1.0  then
Erlen:=-50.5646      + 734.9293*E0 + 830.77527*sqr(E0)
      - 356.37912*sqr(E0)*E0  else
Erlen:=-262.7650      + 1463.058*E0 - 40.92257*sqr(E0)
      + 0.852763*sqr(E0)*E0;
end  else
if cell = -1  then  (* 33 % *)
begin
  if E0 <= 0.2  then
    Erlen:=-9.05805E-1 + 103.2968*E0 + 3661.0080*sqr(E0)
          - 5180.9561*sqr(E0)*E0  else
    if E0 <= 1.0  then
      Erlen:=-50.0565      + 728.0881*E0 + 823.5880*sqr(E0)
            - 353.3844*sqr(E0)*E0  else
      Erlen:=-260.36929      + 1449.811*E0 - 40.6388*sqr(E0)
            + 0.8467635*sqr(E0)*E0;
    end  else
  if cell = 0 then
  begin
    if E0 <= 0.2 then
      Erlen:=-8.95767E-1 + 101.9048*E0 + 3649.2866*sqr(E0)
            - 5.15575E3*E0*sqr(E0)  else
    if E0 <= 1.0 then
      Erlen:=-5.02875E1 + 7.273489E2*E0 + 8.190796E2*sqr(E0)
            - 3.504776E2*E0*sqr(E0)  else
      Erlen:=-2.60305E2 + 1.446344E3*E0 - 3.975610E1*sqr(E0)
            + 8.283120E-1*E0*sqr(E0);
    end  else
  if cell=1 then
  begin
    if E0<= 0.20 then
      Erlen:= -0.9078 + 103.5243*E0 + 3670.3027*sqr(E0)
            - 5193.8482*sqr(E0)*E0      else
    if E0<=1.0  then
      Erlen:= -47.4253 + 707.2341*E0 + 877.5254*sqr(E0)
            - 388.8358*sqr(E0)*E0      else
      Erlen:= -261.023 + 1453.507*E0 - 40.717*sqr(E0)
            + 0.84842*sqr(E0);
    end  else
  if cell=2 then
  begin
    if E0<= 0.20 then

```

```

        Erlens:= -0.8763 + 100.042*E0 + 3528.800*sqr(E0)
              - 4997.51*sqr(E0)*E0      else
if E0<=1.0 then
        Erlens:= -48.067 + 700.994*E0 + 794.971*sqr(E0)
              - 341.500*sqr(E0)*E0      else
        Erlens:= -250.77 + 1397.20*E0 - 39.4954*sqr(E0)
              + 0.82267*sqr(E0)*E0;
end else
if cell=3 then
begin
    if E0<= 0.20 then
        Erlens:= -0.8649 + 99.6798*E0 + 3378.6836*sqr(E0)
              - 4828.662 *sqr(E0)*E0      else
    if E0<=1.0 then
        Erlens:= -46.212 + 678.5045*E0+ 738.37 *sqr(E0)
              - 321.266*sqr(E0)*E0      else
        Erlens:= -230.19 + 1321.85*E0 - 41.826 * sqr(E0)
              + 0.9188*e0*sqr(E0);
    end else
begin
    if E0<= 0.20 then
        Erlens:= -0.96021 + 109.62 *E0 + 3864.6200*sqr(E0)
              - 5481.251 *sqr(E0)*E0      else
    if E0<=1.0 then
        Erlens:= -52.739 + 769.09 *E0+ 864.67 *sqr(E0)
              - 372.675*sqr(E0)*E0      else
        Erlens:= -269.858 + 1522.4*E0 - 43.453*sqr(E0)
              + 0.9246*E0*sqr(E0);
    end;
end;

(* reduced energy vs. penetration length for electron *)

Function EnLeft( xx : real ): real;
var elef, dx : real;
begin
    dx:= Rlen - xx;
    dx:= dx / 1.0E4;
    if ( XX >= rlen ) then Elef:=0.0      else
begin
    if cell = -2 then
begin
    if Ein <= 0.06 then
        Elef:=6.71898E-3 + 55.73670*dx - 25122.9085*sqr(dx)

```

```

        + 6.1416611E6*sqr(dx)*dx   else
    if Ein <= 0.2   then
    Eleft:=2.43462E-2 + 22.39569*dx - 1121.2664*sqr(dx)
        + 3.5767907E4*sqr(dx)*dx   else
    if Ein <= 0.75   then
    Eleft:=7.39327E-2 + 10.6759*dx - 49.637098*sqr(dx)
        + 248.90519*sqr(dx)*dx   else
    Eleft:=1.79908E-1 + 6.89179*dx + 1.37183*sqr(dx)
        + 0.3290142*sqr(dx)*dx;
    end   else
    if cell = -1   then
begin
    if Ein <= 0.06   then
    Eleft:=6.7175E-3 + 56.2156*dx - 25553.5627*sqr(dx)
        + 6.300951E6*sqr(dx)*dx   else
    if Ein <= 0.2   then
    Eleft:=2.4340E-2 + 22.5947*dx - 1141.1113*sqr(dx)
        + 3.672471E4*sqr(dx)*dx   else
    if Ein <= 0.75   then
    Eleft:=7.3879E-2 + 10.7762*dx - 50.5997*sqr(dx)
        + 256.1124*sqr(dx)*dx   else
    Eleft:=1.7984E-1 + 6.95545*dx + 1.39815*sqr(dx)
        + 0.34241*sqr(dx)*dx;
end   else
    if cell = 0   then
begin
    if Ein <= 0.06   then
    Eleft:=6.7295E-3 + 56.6382*dx - 25964.7837*sqr(dx)
        + 6.447600E6*sqr(dx)*dx   else
    if Ein <= 0.2   then
    Eleft:=2.4392E-2 + 22.7070*dx - 1153.9870*sqr(dx)
        + 3.732092E4*sqr(dx)*dx   else
    if Ein <= 0.75   then
    Eleft:=7.4360E-2 + 10.7862*dx - 50.4696*sqr(dx)
        + 255.6569*sqr(dx)*dx   else
    Eleft:=1.8051E-1 + 6.96689*dx + 1.39212*sqr(dx)
        + 0.30856*sqr(dx)*dx;
end   else
    if cell = 1   then
begin
    if Ein <= 0.06   then
    Eleft:=6.7179E-3 + 56.0810*dx - 25432.1632*sqr(dx)
        + 6.255908E6*sqr(dx)*dx   else
    if Ein <= 0.2   then

```

```

Eleft:=2.4341E-2 + 22.5388*dx - 1135.5200*sqr(dx)
      + 3.645419E4*sqr(dx)*dx   else
if Ein <= 0.75 then
Eleft:=7.3895E-2 + 10.7480*dx - 50.3274*sqr(dx)
      + 254.0705*sqr(dx)*dx     else
Eleft:=1.7986E-1 + 6.93756*dx + 1.39073*sqr(dx)
      + 0.33859*sqr(dx)*dx;
end   else
if cell = 2 then
begin
  if Ein <= 0.06 then
    Eleft:=6.7121E-3 + 58.2063*dx - 27383.0354*sqr(dx)
          + 6.992584E6*sqr(dx)*dx   else
  if Ein <= 0.2 then
    Eleft:=2.4316E-2 + 23.4220*dx - 1225.4735*sqr(dx)
          + 4.088511E4*sqr(dx)*dx   else
  if Ein <= 0.75 then
    Eleft:=7.3654E-2 + 11.1937*dx - 54.7031*sqr(dx)
          + 287.5420*sqr(dx)*dx     else
    Eleft:=1.7958E-1 + 7.22016*dx + 1.51002*sqr(dx)
          + 0.40181*sqr(dx)*dx;
  end   else
if cell = 3 then
begin
  if Ein <= 0.06 then
    Eleft:=6.6644E-3 + 59.7865*dx - 28777.7318*sqr(dx)
          + 7.562258E6*sqr(dx)*dx   else
  if Ein <= 0.2 then
    Eleft:=2.4878E-2 + 23.8571*dx - 1240.8724*sqr(dx)
          + 4.212895E4*sqr(dx)*dx   else
  if Ein <= 0.75 then
    Eleft:=7.2921E-2 + 11.7078*dx - 59.0603*sqr(dx)
          + 328.3620*sqr(dx)*dx     else
    Eleft:=1.7471E-1 + 7.63919*dx + 1.86041*sqr(dx)
          + 0.69492*sqr(dx)*dx;
  end   else
if cell = 4 then
begin
  if Ein <= 0.06 then
    Eleft:=6.7115E-3 + 53.1383*dx - 22821.4950*sqr(dx)
          + 5.320835E6*sqr(dx)*dx   else
  if Ein <= 0.2 then
    Eleft:=2.4307E-2 + 21.3896*dx - 1021.8507*sqr(dx)
          + 3.114575E4*sqr(dx)*dx   else

```



```

        Dcy:= -0.78798  + 0.7259  * xx + 0.44538 * sqr(xx)
              - 0.037336* sqr(xx) *xx;
end   else
if cell = 2 then
begin
    if xx < 0.30  then
        Dcy:= 0.0  else
    if xx < 1.1  then
        Dcy:= 0.0466  - 0.52535 * xx + 1.3945 * sqr(xx)
              - 0.43876 * xx * sqr(xx)  else
        Dcy:= -0.97400  + 0.9147  * xx + 0.52926 *sqr(xx)
              - 0.04475 * xx *  sqr(xx);
    end   else
if cell=3 then
begin
    if xx < 0.32 then
        Dcy:=0.0  else
    if xx < 1.1 then
        Dcy:= 0.07027  - 0.7030  *xx  + 1.7493  *sqr(xx )
              - 0.55049  *sqr(xx )*xx  else
        Dcy:= -1.1404 +1.0277  *xx +0.66216  *sqr(xx )-0.057361  *
              sqr(xx )*xx ;
    end   else
begin
    if xx < 0.60 then
        Dcy:=0.0  else
    IF xx < 2.2 THEN
        Dcy:= 0.07911  - 0.2661  *xx  + 0.2598  *sqr(xx )
              - 0.03838  *xx *sqr(xx )  else
        Dcy:= 0.1524  - 0.4971  *xx  + 0.32364  *sqr(xx )
              -0.0251  *sqr(xx )*xx ;
    end;
end;

Function RoX( x : real): real;
begin
    RoX:= absorpt * exp ( - (x) * absorpt);  (* omit (x-lower) *)
end;

Function omnidx(i:integer; x:real):real;
var
    j : integer;
    xx, xy, xz,
    H, x0,

```



```

    x1,x2: real;
begin
    H:= (1.0 - 0.01)/(2*omiter);
    if (particle = proton) then
    x0:= ABS(Dcx(eeleft(Xj[i],2)) - Dcx(eeleft(x,2))) +
        ABS(Dcx(eeleft(xj[i]/0.01,2)) -Dcx(eeleft(x,2)));
    if (particle = electron) then
    x0:= ABS(Dcy(enleft(xj[i])) - Dcy(enleft(x))) +
        ABS(Dcy(enleft(xj[i]/0.01)) - Dcy(enleft(x)));
    x1:= 0.0;
    x2:= 0.0;
    for j:= 1 to (2*omiter - 1) do
    begin
        xx:= 0.01 + h * j;
        if particle = proton then
        begin
            xy:= Dcx(eeleft(xj[i]/xx,2));
            xz:= Dcx(eeleft(x,2));
        end else
        begin
            xy:= Dcy(enleft(xj[i]/xx));
            xz:= Dcy(enleft(x));
        end;
        if odd(j) then
            x1:= x1 + abs(xy - xz) else
            x2:= x2 + abs(xy - xz);
        end;
        Omnidx:= 2*pai*H*(x0 + 2.0*x2 + 4.0*X1) / 3.0;
    end;

function caldcx(x:real; i : integer):real;
var temp, temp1 : real;
    H, x0, xx,
    x1, x2, xi : real;
    M, j : integer;
begin
    if not Omni then
    begin
        if particle = proton then
        begin
            temp:= abs(dcx(eeleft(x,1))-Dcx(eeleft(Xj[i],1)) );
            temp1:=abs(dcx(eeleft(x,2))-dcx(eeleft(Xj[i],2)) );
            Caldcx:= ( temp + temp1 ) * 0.5 ;
        end else

```

```

        Caldcx:= abs(dcy(enleft(x))-Dcy(enleft(Xj[i])));
    end else
        Caldcx:= Omnidx(i,x);
    end;

Function FandRo(x,u : real ; i : integer) : real;
var temp, temp1 : real;
begin
    temp:=caldcx(x, i);
    if x>= Xj[i] then sigmaR:=sigmaH[i]    else
        sigmaR:=sigmaE[i] ;
    FandRo:=exp( -sqrt(6)*sigmaR*temp*FLUX / u) * Rox(x);
end;

Function Photoabsorb (upper : real ): real;
var
    H, x, x0,
    x1,x2, xi      : real;
    M, I, J        : integer;
begin
    H:= ( upper ) / ( 2.0* DIST );
    x0:= RoX(upper) + RoX(0.0);
    x1:=0.0;
    x2:=0.0;
    for i:= 1 to (2*DIST-1) do
        begin
            x:= 0.0 + I*H;
            if odd(i) then
                x1:=x1 + Rox(x)    else
                x2:=x2 + RoX(x) ;
            end;
        XI := H * ( x0 + 2.0* X2 + 4.0* x1 ) / 3.0;
        Photoabsorb:=Xi;
    end;

Function FractionLoss(upper : real ; ii: integer) : real;
var
    Hx, Hu, J1, J2, J3,
    K1, K2, K3, L, Jout,
    Z, x, u          : real;
    I, J, n, m       : integer;
begin
    Hx:=(upper ) / ( 2.0*DIST );
    J1:=0.0; j2:=0.0; j3:=0.0;

```

```

for i:= 0 to ( 2*DIST ) do
begin
  x:= 0.0 + I*Hx;
  Hu:=(1- 0.0) / ( 2.0 * DIST); (* instead of 1E-6 *)
  K1:= 0.0 + FandRo(x,1, ii ); (* FandRo(x,0)=0.0 *)
  k2:=0;
  K3:=0;
  for j:=1 to ( 2*DIST-1 ) do
  begin
    u:=0.0 + j*Hu;
    Z := FandRo(x,u, ii);
    if odd(j) then
      k3:=k3 + z   else
      k2:=k2 + z ;
  end;
  L := ( k1 + 2.0*k2 + 4.0*K3 ) * Hu / 3.0;
  if ( i=0 ) or ( i= (2*DIST)) then
    j1:=j1 + L   else
  if odd(i) then
    j3:=j3 + L   else
    j2:=j2 + L;
  end;
  Jout:=( J1 + 2.0*j2 + 4.0*j3 ) * Hx / 3.0;
  Fractionloss:=Jout;
end;

procedure printout(ij : integer);
var
  i, j : integer;
begin
  writeln(out,Ij, ' ', ReduceC[ij]);
end;

procedure coverglass( i : integer ) ;
var
  temp, temp1 : real;
  ii : integer;
begin
  ii:= cell;
  cell:=0;
  if particle = proton then
  begin
    Plen:=Plength(Ein);
    Rlen:=Rlength(Ein);

```

```

        Ein:=Eeleft(Wd[i],1);
    end;
    cell:= ii;
end;

Function AlGaAs33( temp : real):real;
var
    i, j : integer;
    atemp,
    btemp : real ;
begin
    Atemp:= 1.24 / temp + 0.0667 ;
    if atemp <= 2.0 then
        Btemp:=exp((27.5373*atemp-51.2296)*ln10) / 1E4    else
    if atemp <= 2.2658 then
        Btemp:=exp((1.7156*atemp+0.4138)*ln10) / 1E4    else
    if atemp <= 2.3324 then
        Btemp:=exp((8.1696*atemp-14.2096)*ln10) / 1E4    else
    if atemp <= 2.3999 then
        Btemp:=exp((2.3258*atemp-0.5796)*ln10) / 1E4    else
    if atemp <= 3.1316 then
        btemp:= exp((0.4108*atemp+4.0145)*ln10) / 1E4    else
        Btemp:= 20.0;
    AlGaAs33 := btemp;
end;

```

```

Function AlGaAs40( temp : real):real;
var
    i, j : integer;
    atemp,
    btemp : real ;
begin
    Atemp:= 1.24 / temp ;
    if atemp <= 2.0 then
        Btemp:=exp((27.5373*atemp-51.2296)*ln10) / 1E4    else
    if atemp <= 2.2658 then
        Btemp:=exp((1.7156*atemp+0.4138)*ln10) / 1E4    else
    if atemp <= 2.3324 then
        Btemp:=exp((8.1696*atemp-14.2096)*ln10) / 1E4    else
    if atemp <= 2.3999 then
        Btemp:=exp((2.3258*atemp-0.5796)*ln10) / 1E4    else
    if atemp <= 3.1316 then
        btemp:= exp((0.4108*atemp+4.0145)*ln10) / 1E4    else
        Btemp:= 20.0;

```

```

    AlGaAs40 := btemp;
end;

Function GetAbsorp( II : integer ; temp : real) : real;
begin
    if ii = -2 then
        GetAbsorp:= AlGaAs40(temp)    else
    if ii = 1 then
        GetAbsorp:= AlGaAs33(temp)    else
    if ii = 2 then
        GetAbsorp:= GaAsAbsorp(temp) else
    if ii = 3 then
        Getabsorp:= InGaAsAbsorp(temp) else
    if ii = 4 then
        GetAbsorp:= GeAbsorp(temp)    else
        GetAbsorp:= SiAbsorp(temp);
end;

function degradation( ii, jj : integer): real;
var
    i, j      : integer;
    x0, x00,
    x1, x10,
    x2, x20,
    xi, xi0 : real;
    H        : REAL;
begin
    h:= ( high[jj] - low[jj] ) / ( 2 * Aiter );
    absorpt:=GetAbsorp(ii,low[jj]);
    x0:= fractionloss(UpPene,jj); x00:= photoabsorb(Uppene);
    absorpt:=GetAbsorp(ii,high[jj]);
    x0:=x0 + fractionloss(UpPene,jj ); x00:=x00+ photoabsorb(Uppene);
    x1:=0.0; x2:=0.0; x10:=0.0; x20:=0.0;
    for i:= 1 to ( 2* Aiter -1 ) do
        begin
            absorpt:=GetAbsorp(ii,low[jj]+i*h);
            if odd(i) then
                begin
                    x10:=x10 + photoabsorb(Uppene);
                    x1 := x1 + fractionloss(Uppene,jj );
                end
            else
                begin
                    x2:= x2 + fractionloss(UPPENE,jj ) ;
                    x20:=x20 + photoabsorb(UpPene);
                end
            end
        end
    end

```

```

        end;
    end;
    xi:= H * ( x0 + 2.0*x2 + 4.0*x1 ) / 3.0 ;
    xi0:=H * ( x00 + 2.0*x20 + 4.0*x10 ) / 3.0;
    degradation:=xi / xi0;
end;

Procedure irradiated;
var i, j :integer;
    beg,
    endd : integer;
    ch : char;
    STOP  : BOOLEAN;
    TEMPE ,
    REFIRST,
    RESECONd,
    REDIFF : REAL;
begin
    WRITELN(OUT,' ENERGY LEVEL ? ',EIN );
    for i:= 1 to Config do
        begin
            Tempe:= Ein;
            cell:= Name[i];
            Stop := false;
            dist:= 4;
            Refirst:= 0.0;
            Resecon d:= 0.0;
            repeat
                coverglass(i);
                Dxij:= Xj[i];
                if particle = proton then
                    Rlen:= Rlength(ein) else
                    Rlen:= Erlen(ein);
                if rlen <= Tj[i] THEN
                    UpPene := Rlen ELSE
                    UPPENE:= Tj[i] ;
                Resecon d:= degradation(cell,1);
                if Abs(Resecon d - Refirst) <= 0.000001 then
                    Stop := true else
                    begin
                        Dist:= Dist + 1 ;
                        Ein := Tempe;
                        Refirst:= Resecon d ;
                    end;
            until Stop;
        end;
    end;
end;

```

```

        if dist >= 100 then
            Stop := true;
        until stop ;
        ReduceC[i]:= Refirst;
        writeln(out,' Iteration = ',dist);
        printout(I);
        if Tj[i] > Rlen then
            writeln(out,' Only cell #',cell,' .. degrades') else
            begin
                if particle = proton then
                    Ein:= eeleft(Tj[i],1) else
                    Ein:= enleft(Tj[i]);
                end;
            end;
        end;
    end;
end;

(* Irradiation calculation for space environment,proton *)

Procedure SpaceProton;
var
    i, j : integer;
    t      : real;
begin
    for i:=1 to 9 do
        begin
            for j:=1 to 3 do
                begin
                    if i=1 then begin ein:=1E5; t:=1.2E14; end else
                    if i=2 then begin ein:=2E5; t:=6.2E13; end else
                    if i=3 then begin ein:=4E5; t:=2.0E13; end else
                    if i=4 then begin ein:=1E6; t:=2.2E12; end else
                    if i=5 then begin ein:=2E6; t:=3.7E13; end else
                    if i=6 then begin ein:=3E6; t:=1.1E13; end else
                    if i=7 then begin ein:=4E6; t:=7.5E12; end else
                    if i=8 then begin ein:=6E6; t:=6.7E11; end else
                    if i=9 then begin ein:=1E7; t:=9.2E10; end;
                    if j=1 then flux:=t* 3   else
                    if j=2 then flux:=t*7   else
                        flux:=t*10;
                    writeln(out,'ein= ',ein,' flux= ',flux);
                    irradiated;
                end;
            end;
        end;
    end;
end;

```

```

procedure SpaceElectron;
var
  i, j : integer;
  t : real;
begin
  for i:=1 to 7 do
    begin
      for j:=1 to 3 do
        begin
          if i=1 then begin ein:=1E5; t:=7.4E14; end else
          if i=2 then begin ein:=5E5; t:=6.4E13; end else
          if i=3 then begin ein:=1E6; t:=1.8E13; end else
          if i=4 then begin ein:=2E6; t:=3.4E12; end else
          if i=5 then begin ein:=3E6; t:=6.1E11; end else
          if i=6 then begin ein:=4E6; t:=8.5E10; end else
          begin ein:=5E6; t:=8.5E9 ; end;
          if j=1 then flux:=t*3   else
          if j=2 then flux:=t*7   else
          flux:=t*10;
          writeln(out,'ein= ',ein,' flux= ',flux);
          irradiated;
        end;
      end;
    end;
end;

(* Calculations for damage constants of minority carrier diffusion *)
(* length and minority carrier lifetime. *)

Procedure damage;
const
  k = 1.38E-23;
var
  Ans : char;
  Index : integer;
  Rplen : real;

procedure getmass( ii : integer);
var
  i : integer;
  percent,
  MpTemp,
  MpH,
  MpL : real;

```



```

begin
  writeln(' what is the percentage of X (A1) ? ');
  readln(percent);
  MnEff[ii]:= 0.067 + 0.083 * percent;
  MpL:= 0.082 + 0.063 * percent;
  MpH:= 0.62 + 0.14 * percent;
  MpTemp:= exp(1.5*ln(MpL)) + exp(1.5*ln(MpH));
  MpEff[ii]:= exp(2/3.0 * ln( MpTemp));
end;

procedure getVth(ii : integer);
var
  i : integer;
  TT,
  Vth1,
  Vth2 : real;
begin
  writeln(' what is the Temperature ? ');
  readln(TT);  writeln(out, ' TT = ',TT);
  VthN[ii]:= sqrt( 3* K * TT / (MnEff[ii]*Me)) * 100 ;
  VthP[ii]:= sqrt( 3* K * TT / (MpEff[ii]*Me)) * 100 ;
end;

procedure Dinitia1;
var
  i, j : integer;
  ch,
  ans : char;
begin
  for i:= 1 to Config do
    begin
      writeln( i:1, ', Ln and Lp ? um ');
      readln(Lnn[i],Lp[i]);
      writeln(out, ' Lnn[' ,i:1,']=',Lnn[i], ' Lp[' ,i:1,']=',Lp[i]);
      writeln( i:1, ', Tn and Tp ? sec '); readln(TauN[i],TauP[i]);
      writeln(out, ' Tn[' ,I:1,']=',TauN[i], ' Tp[' ,i:1,']=',TauP[i]);
      GetMass(config); writeln(out, 'Mn = ',Mneff[i], ' Mp = ',Mpeff[i]);
      GetVth(Config);
      writeln(out, ' VthN (cm) = ',VthN[i], ' VthP(cm) = ',VthP[i]);
      Dpp[i]:= Lp[i] * Lp[i] * 1E-8 / TauP[i];
      Dn[i] := Lnn[i] * Lnn[i] * 1E-8 / TauN[i];
      writeln(out, ' Dn = ',Dn[i], ' Dpp = ', Dpp[i]);
    end;
  end;
end;

```

```

Function geteleleft(ii : integer; xx : real): real;
begin
  cell := ii;
  if particle = electron then
    geteleleft:= enleft(xx)    else
    GetEleleft:=(Eeleleft(xx,1) + EELEFT(XX,2))/2.0;
end;

Function GetDefect( ii : integer; xx : real) : real;
begin
  cell := ii;
  if particle = electron then
    GetDefect:= Dcy(xx) else
    GetDefect:= Dcx(xx);
end;

Function GetLen( ii : integer; xx : real):real;
begin
  cell := ii;
  if particle = electron then
  BEGIN
    rLEN := Erlen(xx) ;
    gETLEN := RLEN;
  END ELSE
  BEGIN
    RLEN:= RLENGTH(XX);
    PLEN:= PLENGTH(XX);
    GetLen:= (rlen + PLEN)/2.0;
  END;
end;

Function GetConst(ii, jj : integer): real;
var
  Kl,
  Kt : real;
begin
  if ii = 1 then
  begin
    Kt := ( 1 / TauNIrr[jj] - 1 / TauN[jj] ) / flux ;
    Kl := Kt / Dn[jj];
  end else
  begin
    Kt := ( 1 / TauPIrr[jj] - 1 / TauP[jj] ) / flux ;

```

```

        Kl := Kt / Dpp[jj];
    end;
    GetConst:= Kl;
end;

procedure GetDamage(ii: integer);
var
    i : integer;
    Etemp : real;
    R1, R2,
    Nt,
    Nt1,
    Nt2 : real;
begin
    if window[ii] then
        begin
            RPlen := getLen(0,Ein);
            Ein:= getEleft(0,Wd[ii]);
        end;
        i := Name[ii];
        RPlen:= getLen(i,Ein);
        Nt1:= GetDefect(i,Ein);
        writeln(out,' defect ## ',Nt1, ' Ein = ',Ein);
        if RPlen <= Xj[ii] then
            begin
                Nt:= Nt1 / RPlen * 1E4;
                Etemp:= 0.0;
            end
        else
            begin
                Etemp:= getEleft(i,Xj[ii]);
                Nt2:= getDefect(i,Etemp);
                writeln(out,' ### 2 ',nt2, ' Ein = ',Etemp);
                Nt := ( Nt1 - Nt2 ) / Xj[ii] * 1E4;
            end;
        TauNIrr[ii]:= 1 / ( SigmaE[ii] * Nt * Flux * VthN[ii]);
        Kln[ii]:=Getconst(1,ii);
        if Etemp >= 0 then
            begin
                Ein := Etemp;
                RPlen:= getLen(i,Ein);
                Nt1:= GetDefect(i,Ein);
                writeln(out,' $$$$ ',Nt1,' Ein = ',Ein);
                if RPlen <= Tj[ii] then
                    begin

```

```

        Nt:= Nt1 / RPlen * 1E4;
        Etemp:= 0.0;
    end else
    begin
        Etemp:= getEleft(i,Tj[ii]);
        Nt2:= getdefect(i,Etemp);
        Nt := ( Nt1 - Nt2 ) / Tj[ii] * 1E4 ;
        writeln(out,'$$$0 ',Nt2,' Ein =',etemp);
    end;
    TauPIrr[ii]:= 1 / ( sigmaH[ii] * Nt * flux * VthP[ii]);
    Klp[ii]:= Getconst(2,ii) ;
    end else
    Klp[ii]:= 0.0;
    Ein:= Etemp ;
end;

begin (* damage *)
    writeln(' what kind of particle ? Proton or Electron ');
    readln(particle);
    writeln(' Ein (eV) = ');
    readln(Ein); writeln(out,' Ein = ',ein);
    writeln(' what is the fluence of the particle ? ');
    readln(flux); writeln(out,' Flux = ',flux);
    Dinitia;
    repeat
    for index:=1 to config do
    begin
        RPlen:= getlen(index,Ein);
        getDamage(index);
        writeln(' Kln = ',Kln[index], ' Klp = ',Klp[index]);
        writeln(out,' Kln = ',Kln[index], ' Klp = ',Klp[index]);
    end;
    writeln(' more Ein to run ? "y" or "n" ');
    readln(Ans);
    if (Ans = 'y') or (Ans = 'Y') then
        Runn :=true           else
        Runn :=false;
    if runn then
    begin
        writeln(' Ein (eV) = ? '); readln(Ein);
        writeln(out); writeln(out,'Ein = ', Ein);
    end;
    Until ( not Runn )
end; (* damage *)

```

```

begin (* of main *)
  writeln(' --> solar cell starts !      Running sequence ?');
  writeln(' 1 = calculation for specified Ein and Flux for Proton. ');
  writeln(' 2 = Specified Ein and Flux for electron ');
  writeln(' 3 = Space environment for Proton ');
  writeln(' 4 = Space environment for electron ');
  writeln(' 5 = Calculations of damage constant ');
  readln(sequence);
  writeln(' Absorption Coeff. iteration ,10< Aiter < 25 ');
  readln(aiter);
  initial;
  case sequence of
    1 : begin
        particle:=proton;
        writeln(' specified the Ein and Flux please ');
        writeln(' Ein <= 1E7 eV; Flux <= 1E12 ');
        readln(Ein,flux);
        if ein <= 10E6 then
          irradiated;
        end;
    2 : begin
        particle:=electron;
        writeln('specified the Ein and Flux please ');
        writeln(' Ein <= 5E6; Flux <= 1E16 ');
        readln(Ein,flux);
        if ein<=5.0E6 then
          irradiated;
        end;
    3 : begin
        particle:=proton;
        SpaceProton;
        end;
    4 : begin
        particle:=electron;
        SpaceElectron;
        end;
    5 : begin
        Damage;
        end;
  end; (* case *)
end.

```

APPENDIX C

AN EXPERT SYSTEM PROGRAM FOR OPTIMAL DESIGN OF SINGLE-JUNCTION AND MULTIJUNCTION TANDEM SOLAR CELLS

```
{
This program is the kernel of the expert system. A modified Box complex
search optimization algorithm has been implemented in this program. The
rule-based has been implemented as pattern-directed module. They are in
the form of if-then format. This program can integrate the program in
Appendix B into a large system. However, because it takes a little while
to calculate the degradation of the irradiated solar cell, this program
will take the results from Appendix B namely the damage constant as its
input for optimizing the cell design of irradiation case. This program
is user friendly. the purpose of each procedure is described by the name
of the proceudre itself. The program is run in Harris 800 computer system.
}
```

```
Program optima(Input, output, Inp1(lfn=103), Inp2(lfn=104),
               Inp3(lfn=105), Inp4(lfn=106), Out(lfn=107), inp5(lfn=108));
```

```
Const
```

```
Qe      = 1.602E-19;
Pemit   = 8.854E-14;
ln10    = 2.30258509;
```

```
TYPE
```

```
IndType = array [1..20] of real;
IntType = array [1..20] of integer;
ArrayType = array [1..20, 1..20] of real;
InpType = array [1..200] of real;
DataType = array [1..3] of real;
KvType = array [0..40] of real;
CellType = (ALGAAS, GE, SI, GAAS, INGAAS);
RGType = (RECOMBINATION, DIFFUSION);
```

```
Var
```

```
Inp1, Inp2,
Inp3, Inp4,
out, Inp5      : text;
AdaVec,
PreGuess,
```

```

Guess,
AveX,
Ubound, Lbound: Indtype;
InitGuess      : ArrayType;
IndVar,
Wfactor,
CellConf,
CellNum,
Step,
IAR,
Iter,
Insolation,
Particle,
TunnelNum,
AirM           : integer;
ColEff,        (* Collecting Efficiency *)
IdealF,        (* Ideality Factor *)
KLn, KLp,      (* damage constant *)
Na, Ndd,       (* doping density *)
Joo,           (* dark current density *)
Voc,          (* open circuit voltage *)
Vm,           (* maximum open circuit voltage *)
FF,           (* fill factor *)
Ada,          (* conversion efficiency *)
Aj,           (* cell area *)
Eg,           (* bandgap energy *)
Sn, Sp,       (* surface velocity *)
Mobe, Mobp,   (* mobilities *)
Lnn, Lpp,     (* minority diffusion length *)
Dnn, Dpp,     (* diffusion constant *)
Tn, Tp,       (* minority carrier lifetime *)
Xj, Tj,       (* junction depth, cell thickness*)
Mn, Mp,       (* effective mass *)
Wj, Ni        : DataType; (* depletion width *)
Airmass,
Pam,          (* Solar Irradiance *)
Rs, Rs1,      (* Series Resistance *)
LossRatio,
ReFL, Shadow, (* losses *)
Fluxx,        (* Particel fluences *)
Fworst, Fbest,
Tratio,       (* T / 300 *)
AdaTot,
High, Low,

```

```

Jsc ,
error, TuRatio,
Xial, Alratio,
Teff,
Tdegree      : real;
CellName     : array [1..3] of Celltype;
RGmodel      : array [1..3] of RGType;
Atype        : inptype;
Btype        : array [1..6] of inptype;
WL1, Kv1,
WL2, Kv2,
WL3, Kv3     : KvType;
Tna, Tnd     : array [1..2] of real; (* doping density of tunnel Jn *)
System,
Mic,
Btunnel,
PN,
GFactor      : boolean;
Fxy, Fxx, Fyy : InpType;
TunnelCell,
TunnelTd,
nindex,
TdropV,
ARtd         : array [1..2] of real;

```

```

Procedure Initial;

```

```

var

```

```

    i, j : integer;

```

```

begin

```

```

    Mic:= false;

```

```

    Gfactor:= true;

```

```

    PN := true;

```

```

    Alratio:= 0.00;

```

```

    for i:= 0 to 23 do

```

```

        readln(inp1,WL1[i],Kv1[i]); (* for GaAs *)

```

```

    for i:= 0 to 35 do

```

```

        readln(inp2,WL2[i],Kv2[i]); (* for Si  *)

```

```

    for i:= 0 to 11 do

```

```

        readln(inp3,WL3[i],Kv3[i]); (* for Ge  *)

```

```

    for i:= 1 to 196 do

```

```

        begin

```

```

            for j:= 1 to 6 do

```

```

                read(inp4,Btype[j,i]);

```

```

            Atype[i]:= 1E4 / (1000 + (I-1) * 200);

```



```

end;
for i:= 1 to 168 do
    readln(INP5, x1a1, Fxy[i]);
Fxx[1]:= 0.099; Fxx[2]:= 0.198; Fxx[3]:= 0.315; Fxx[4]:= 0.419;
Fxx[5]:= 0.491; fxx[6]:= 0.590; Fxx[7]:= 0.700; Fxx[8]:= 0.804;
for i:= 1 to 21 do
    Fyy[i]:= 1.5 + (i-1)*0.1;
end; (* of initial *)

```

Procedure ReadInput;

var

RG,

I : integer;

ch : char;

begin

```

writeln(' Welcome to the solar cell expert system. ');
writeln(' You could have at most 3 P/N Junctions in your design. ');
writeln(' How many p/n junctions you want for this run ? ');
readln(CellConf);
writeln(' At what Temperature? ');
readln(tdegree);
Tratio:= Tdegree / 300;
writeln(' At what kind of Airmass? ');
writeln(' 0 : For space application ');
writeln(' 1, 1.5, 2.0, 2.5, 3.0 for terrestrial application ');
readln(Airmass);
if (Round(Airmass*10) = 0 ) then
begin
    AirM:= 1;
    Pam:= 135.3;
end
else
if (Round(Airmass*10) = 10) then
begin
    AirM:= 2;
    Pam:= 103.87
end
else
if (Round(Airmass*10) = 15) then
begin
    AirM:= 3;
    Pam:= 97.39;
end
else
if (Round(Airmass*10) = 20) then
begin
    AirM:= 4;

```

```

        Pam:=91.94 ;
    end    else
    if (Round(Airmass*10) = 25) then
    begin
        AirM:= 5;
        Pam:= 87.32;
    end    else
    if (Round(Airmass*10) = 30) then
    begin
        AirM:= 6;
        Pam:= 83.2 ;
    end;
    writeln(' At what kind of sun insolation?');
    readln(insolation);
    writeln(' Cell materials selection. ');
    writeln(' If answer (y) or (Y) then user assign cell materials');
    writeln(' Otherwise, system default optimum materials');
    readln(ch);
    if (ch = 'y') or (ch = 'Y') then
        system:= false           else
        system:= true;
end;

Procedure CellSelection;
var
    RG,
    i : integer;
begin
    for i:= 1 to cellConf do
    begin
        writeln(' Cell Name [',i:2,']?');
        writeln('1: AlGaAs; 2: GaAs; 3: InGaAs; 4: Ge; 5: Si');
        readln(cellnum);
        if CellNum = 1 then
            CellName[i]:= ALGAAS    else
        if CellNum = 2 then
            CellName[i]:= GAAS      else
        if CellNum = 3 then
            CellName[i]:= INGAAS    else
        if CellNum = 4 then
            CellName[i]:= GE        else
            CellName[i]:= SI;
        writeln('Front surface recombination velocity Sp[',i:2,'] ?');
        readln(Sp[i]);
    end;
end;

```

```

        writeln('Back surface recombination velocity Sn [' ,i:2,' ] ?');
        readln(Sn[i]);
        writeln(' R-G Model [' ,i:2,' ] ?');
        writeln(' 1 : Diffusion model;    2 : Recombination model');
        readln(RG);
        if RG <> 1 then
            RG := 2;
        if RG = 1 then
            begin
                RGmodel[i]:= DIFFUSION;
                IdealF[i]:= 1.0;
            end
            else
                begin
                    RGmodel[i]:= RECOMBINATION;
                    writeln(' Input Ideality Factor; > 1.0 ');
                    readln(IdealF[i]);
                end;
            end;
        end;
    end;

Procedure InputBound;
var
    i, j : integer;
begin
    if not system then
        begin
            writeln(' Input # of independent variables ');
            writeln(' 1 junction ---> IndVar = 5 ');
            writeln(' 2 junction ---> IndVar = 10');
            writeln(' 3 junction ---> IndVar = 15');
            readln(IndVar);
        writeln(' Input upper bound and lower bound of the Indep. variables ');
        for I:= 1 to IndVar do
            begin
                if (I = 1) or (I = 6) or (i = 11) then
                    writeln('Input upper and lower bound of Nd. ? = Log(Nd)')
                else
                    if (i = 2) or (I = 7) or (i = 12) then
                        writeln('Input upper and lower bound of Na. ? = Log(Na)')
                    else
                        if (i = 3) or (i = 8) or (i = 13) then
                            writeln('Input upper and lower bound of Junction depth,um')
                        else
                            if (i = 4) or (i = 9) or (i = 14) then

```

```

        writeln('Input upper and lower bound of cell thickness,um')
    else
        begin
            writeln('Input upper bound and lower bound of Eg, eV');
            writeln('Usually Ubound = LBound, Unless AlGaAs');
            end;
            readln(Ubound[i],Lbound[i]);
            writeln(Ubound[i],Lbound[i]);
        end;
    end; (* system *)
end; (* of inputbound *)

Procedure TunnelSelection;
var
    i : integer;
begin
    Btunnel:= false;
    if CellConf >= 2 then
        begin
            writeln(' Do you need tunnel junctions in your design?');
            writeln(' 0 : Mic ; 1 : 1 tunnel junction ');
            writeln(' 2 : 2 tunnel junctions. ');
            writeln(' 3 : MIC + 1 tunnel Junction ');
            readln(TunnelNum);
            if TunnelNum >=3 then
                TunnelNum := 3      else
            if TunnelNum <= 0 then
                TunnelNum:= 0;
            if tunnelNum = 3 then
                begin
                    Mic:= true;
                    TunnelNum:= 1;
                end
            else
                Mic:= false;
            if tunnelNum <> 0 then
                Btunnel:= true      else
                Mic:= true;
            for i:= 1 to TunnelNum do
                begin
                    writeln(' Input Tunnel junction. ');
                    writeln(' 1 : AlGaAs, x= ?% , 2 : GaAs ');
                    readln(TunnelCell[i]);
                    if TunnelCell[i] = 1 then
                        begin

```

```

        writeln(' what is the Al ration, Al >= 0.45 ');
        readln(Turatio);
    end else
        TunnelCell[i]:= 2;
        writeln(' what are Nd and Na of tunnel jn? >= 1E19/cm2');
        readln(Tnd[i], tna[i]);
    end;
end;
end;
If Mic then
begin
    Shadow:= shadow * 2.0;
    Rs:= Rs * 2 + Rs1;
end;
end;

Procedure contact;
Var
    N, Nline,
    ConNum, TypeCon,
    ShapeNum      : integer;
    Wd, lth,
    W, A, B,
    d, h,
    RhoS, RhoF, Rci : real;
    Again          : Boolean;
begin
    writeln(' what are the contact structures ?');
    writeln(' 1 : regular front and back contacts,');
    writeln(' 2 : metal interconnect contact. ');
    readln(ConNum);
    writeln(' P-type or N-type contact for front ?');
    writeln(' 1 : P-type, i.e. AuZn; 2: N-type, Au/Ge/Ni ');
    readln(typecon);
    if typecon = 1 then
    begin
        writeln(' Au/Zn ---> 1200 and 600 Angstrons respectively ');
    end else
    begin
        writeln(' Au/Ge/Ni ----> 1200, 400, 600 Angstrons respectively');
    end;
    writeln(' What is the grid structure ?');
    writeln(' 1 : triangle shape; 2 : rectangle shape .');
    readln(shapeNum);
    writeln(' what is the area of the solr cell? Width x Length, cm2 ');

```

```

readln(Wd,lth);
if shapenum = 1 then
begin
  repeat
    writeln(' What is the width of the triangle?  cm please. ');
    writeln(' Width can not > ',Wd,' (cell width) cm');
    readln(W);
    writeln(' Input # of grid line ');
    readln(Nline);
    N := Trunc(Wd / (2 * W));
    if Nline > N then
    begin
      again:= true;
      writeln(' too many grid lines');
      writeln(' can not > ',N, ' lines for grid width = ',w);
    end
    else
      again := false;
  until (not again);
  writeln(' Input finger length please. Must be < ',lth, 'cm');
  readln(a);
  if a > lth then
    a:= lth;
  RhoF:= 2E-6;
  Rc1:= 1E-4;
  Rhos:= 4E-2;
  Rs:= Rhos + Rc1/sqr(1 - Nline*w) * (1/(12*sqr(Nline)) -
    w / (4*Nline) + sqr(w) / 3 - Nline * sqr(w) * w / 6)
    + RhoF / (4 * W * Nline);
  shadow:= Nline * W * a / 2 / (Wd * lth);
end
else
begin
  repeat
    writeln(' Input the # of fingers ');
    readln( Nline );
    writeln(' what is the finger spacing? cm ');
    writeln(' It is usually <= ',Wd/Nline,' cm');
    readln(b);
    writeln(' what is the Finger width? cm ');
    readln(w);
    N:= trunc(Wd / ( b + w ));
    if Nline > N then
    begin
      again:= true;
      writeln(' too many grid line');
    end
  until (not again);
end

```

```

        writeln(' can not > ',N, ' lines');
    end   else
    again := false;
until (not again);
    writeln(' what is the finger height? cm  ');
    readln(h);
    writeln(' what is the finger length? cm; < Cell Length');
    readln( a);
    writeln(' what is the thickness of window or emitter layer');
    readln(d);
    Rhos:= 4E-2;
    Rhof:= 2E-6;
    Rc1:= 1E-4;
    Rs:= Rhos * sqr(b)/(12*d) + b / W * Rhof / (3*h) * sqr(a)
        + b / w * Rc1;
    Rs1:= Rhos * sqr(b)/(12*0.001) + b / W * Rhof / (3*h) * sqr(a)
        + b / w * Rc1;
    shadow:= Nline * W * a / (Lth * Wd);
end;

end;

Procedure ARcoating;
Var
    ArMat,
    I : integer;
begin
    writeln(' what kinds of AR structure ');
    writeln(' 0 : No AR coating; 1 : single layer; 2 : Double layer ');
    readln(IAR);
    if IAR >= 2 then
        IAR := 2;
    if IAR <= 0 then
        IAR:= 0;
    if IAR <> 0 then
        begin
            for i:= 1 to IAR do
                begin
                    writeln(' what kinds of Coating Materials?');
                    writeln(' 1 : Ta2O5 (2.15); 2 : TiO2 (2.15) ');
                    writeln(' 3 : Si3N4 (1.93); 4 : MgF2 (1.38) ');
                    writeln(' 5 : SiO2 (1.80) 6 : ZnS (2.30) ');
                    readln(ArMat);
                    if ArMat = 1 then
                        nindex[i]:= 2.15
                    else

```

```

        if ArMat = 2 then
            nindex[i]:= 2.15           else
        if ArMat = 3 then
            nindex[i]:= 1.93           else
        if ArMat = 4 then
            nindex[i]:= 1.38           else
        if ArMat = 5 then
            nindex[i]:= 1.80           else
            nindex[i]:= 2.30;
    end;
    if IAR = 1 then
        RefL:= sqr((sqr(nindex[1])-3.0522)/(sqr(nindex[1])+3.052)) else
        RefL:= sqr((sqr(nindex[1])*3.0522 - sqr(nindex[2]))/
        (sqr(nindex[1])*3.0522 + sqr(nindex[2])));
    end else
        RefL:= 0.0;
end;

Procedure ARthickness;
var
    i : integer;
begin
    if IAR <> 0 then
        begin
            for i:= 1 to IAR do
                begin
                    if I = 1 then
                        ARtd[i]:= (1.24/(Eg[1] + High))/(4 * Nindex[i]) else
                        ARtd[i]:= (1.24/(Eg[1] + High))/(4 * Nindex[i]);
                    writeln(out,'AR coating thickness,[ ',I:2,' ' ,ARtd[i], ' um');
                end;
            end;
        end;
end;

Procedure HighInsolation;
Var
    i : integer;
    Jscx, Vocx : real;
begin
    for i:= 1 to CellConf do
        begin
            Jscx:= Jsc * Insolation;
            Vocx:= Voc[i] + 0.0259 * Tratio * ln(Insolation);
            Ada[i]:= ada[i] * Vocx / Voc[i];
        end;
    end;
end;

```



```

        Voc[i]:= Vocx;
    end;
    adatot:= 0.0;
    for i:= 1 to CellConf do
        adatot:= ada[i] + adatot;
    end;

Procedure Tunnel;forward;

Procedure RealAda;
var
    VocT,
    Du, Vr,
    DV : real;
    Vratio,
    Fratio : array [1..3] of real;
    i      : integer;
begin
    if Btunnel then
        tunnel;
        LossRatio:= (1 - RefL) * (1 - Shadow);
        Jsc:= Jsc * LossRatio;
        DV := Jsc * Rs / CellConf;
        VocT:= 0.0;
        for i:= 1 to CellConf do
            begin
                Vratio[i]:= (Voc[i] - DV) / Voc[i];
                Fratio[i]:= 1 - Dv / Voc[i];
                Voc[i]:= Voc[i] * Vratio[i];
                VocT:= VocT + Voc[i];
                FF[i]:= FF[i] * Fratio[i];
                ada[i]:= ada[i]*vratio[i]*Fratio[i]*LossRatio;
            end;
        if Btunnel then
            begin
                Du:= 0.0;
                for i:= 1 to TunnelNum do
                    Du:= Du + TdropV[i];
                Vr:= 1 - Du / VocT;
                if Vr <= 0.0 then
                    Vr:= 0.0;
                end else
                    Vr:= 1.0;
                adatot:= 0.0;
            end
        end
    end
end

```

```

    for i:= 1 to CellConf do
        adatot:= adatot + ada[i];
        AdaTot:= adatot * Vr;
    end;

Function TunWd(I:integer; ratio, Teg : real):real;
var
    Tmn, Tmp, Tni,
    Tvbi, Eps, Tnn, Tx1,
    Tlh, Thh : real;
begin
    Eps:= 12.91 * (1 - ratio) + 10.06 * ratio;
    Tmn:= 0.067 + 0.083 * ratio;
    Tlh:= 0.087 + 0.063 * ratio;
    thh:= 0.62 + 0.14 * ratio;
    tmp:= exp(2/3*ln(Tlh*sqrt(tlh) + thh*sqrt(thh)));
    Teff:= Tmn * Tmp / (Tmn + Tmp);
    Tx1:= Tmn * Tmp ;
    Tni:= sqrt(Tx1*sqrt(tx1)*2.33E31)*tdegree*sqrt(tdegree) *
        exp(-Teg/(0.0259*2*tratio));
    Tnn:= (Tna[i]/1E9)*(Tnd[i]/1E9)/(Tna[i] + Tnd[i]) * 1E18;
    Tvbi:= 0.0259*tratio *ln((Tna[i]/Tni)*(Tnd[i]/Tni));
    TunWd:= sqrt(2*Eps*Pemit/Qe * Tvbi / Tnn) / 1E-8 ;
end;

Procedure Tunnel;
var
    Tw, TEg1,
    TEg, Tmass : real;
    I : integer;
begin
    for i:= 1 to TunnelNum do
        begin
            if TunnelCell[i] = 1 then
                begin
                    if Turatio > 0.45 then
                        Teg1:= 1.424 + 1.247*Turatio + 1.147*sqr(Turatio-0.45) else
                        Teg1:= 1.424 + 1.247*Turatio;
                    TEg:= Teg1 - 2.9725E-4 * (Tdegree - 300);
                end else
                begin
                    TEg:= 1.519 - 5.405E-4*sqr(Tdegree)/(204+Tdegree);
                    turatio:= 0.0;
                end;
            end;
        end;
    end;
end;

```

```

    Tw:= TunWd(i, Turatio, Teg);
    TdropV[i]:= Jsc / (1E10 * sqrt(Teg * Teff) / Tw * exp(
      - 0.40 * sqrt(Teg * Teff) * Tw ));
  end;
end;

Procedure Irradiation;
var
  Ch : char;
  I  : integer;
begin
  writeln('Do you consider the irradiation case?');
  writeln(' (Y)es, (y)es, otherwise NO ');
  readln(ch);
  if (ch = 'Y') or (ch = 'y') then
    begin
      writeln('what kind of particle?');
      writeln(' 0 : no irradiation; 1 : proton; 2 : electron');
      readln(particle);
      if particle <> 1 then
        particle := 2;
      writeln(' what is the fluence?');
      writeln(' Proton : 1E10 to 1E12; Electron: 1E14 to 1E16 ');
      readln(fluxx);
      writeln(' what is the damage constants for Ln and Lp');
      writeln(' Proton   : Ln: order of 1E-8; Lp: 1E-9');
      writeln(' Electron : Lp: 1E-7       ; Lp: 1E-8');
      for i:= 1 to CellConf do
        begin
          writeln(' KLn [' ,I:2,'] and KLP [' ,I:2,']');
          readln(Kln[i],Klp[i]);
        end;
      end else
        particle:= 0;
    end;
end;

Function Bandgap(Name : CellType):real;
begin
  if Name = GAAS then
    Bandgap:= 1.519 - 5.405E-4*sqrt(Tdegree)/(Tdegree + 204) else
  if Name = SI    then
    Bandgap:= 1.170 - 4.73E-4*sqrt(Tdegree)/(Tdegree + 636) else
  if Name = GE    then
    Bandgap:= 0.7437 - 4.77E-4*sqrt(Tdegree)/(Tdegree + 235) else

```

```

if Name = INGAAS then
    Bandgap:= 0.812 - 3.26E-4*Tdegree + 3.31E-7 *sqr(Tdegree) else
if Name = ALGAAS then
    Bandgap:= 1.424 + 1.247*alratio - (3.95 - 1.15*alratio)*1E-4*
        (Tdegree - 300.0);
end;

Procedure GetTopBound;
begin
    Eg[1]:= Bandgap(Cellname[1]);
    PN:= true;
    Sn[1]:= 1E4;
    Sp[1]:= 1E4;
    If Cellname[1] = ALGAAS then
    begin
        Ubound[1]:= 16.0;
        Lbound[1]:= 14.5;
        Ubound[2]:= 17.0;
        Lbound[2]:= 15.5;
    end else
    begin
        Ubound[1]:= 17.5;
        Lbound[1]:= 17.0;
        Ubound[2]:= 18.5;
        Lbound[2]:= 18;
    end;
    Ubound[3]:= 0.5;
    Lbound[3]:= 0.05;
    Ubound[4]:= 10.0;
    Lbound[4]:= 2.0;
    Ubound[5]:= Eg[1];
    Lbound[5]:= Eg[1];
    IdealF[1]:= 1.0;
    RGmodel[1]:= DIFFUSION;
end;

```

```

Procedure GetMedBound;
begin
    Eg[2]:= Bandgap(Cellname[2]);
    PN:= true;
    Sn[2]:= 1E4;
    Sp[2]:= 1E4;
    Ubound[6]:= 17.5;
    Lbound[6]:= 17.0;

```

```

Ubound[7]:= 18.5;
Lbound[7]:= 18.0;
Ubound[8]:= 0.5;
Lbound[8]:= 0.05;
Ubound[9]:= 10.0;
Lbound[9]:= 2.0;
Ubound[10]:= Eg[2];
Lbound[10]:= Eg[2];
IdealF[2]:= 1.0;
RGmodel[2]:= DIFFUSION;
end;

Procedure PruneSearch;
begin
  if CellConf = 1 then
    begin
      CellName[1]:= GAAS;
      GetTopBound;
      IndVar:= 5;
    end
  else
    if CellConf = 2 then
      begin
        CellName[1]:= ALGAAS;
        GetTopBound;
        Ubound[5]:= 1.98;
        Lbound[5]:= 1.85;
        CellName[2]:= GAAS;
        GetMedBound;
        IndVar:= 10;
      end
    else
      begin
        CellName[1]:= ALGAAS;
        GetTopBound;
        Ubound[5]:= 1.98;
        Lbound[5]:= 1.85;
        CellName[2]:= GAAS;
        GetMedBound;
        CellName[3]:= INGAAS;
        Eg[3]:= Bandgap(Cellname[3]);
        PN:= true;
        Sn[3]:= 1E4;
        Sp[3]:= 1E4;
        Ubound[11]:= 17.7 ;
        Lbound[11]:= 17.7;
      end
    end
  end
end;

```

```

    Ubound[12]:= 17.3;
    Lbound[12]:= 17.3;
    Ubound[13]:= 0.5;
    Lbound[13]:= 0.05;
    Ubound[14]:= 10.0;
    Lbound[14]:= 2.0;
    Ubound[15]:= Eg[3];
    Lbound[15]:= Eg[3];
    IndVar:= 15;
    IdealF[3]:= 1.0;
    RGmodel[3]:= DIFFUSION;
end;
end; (* PruneSearch *)

Function GaAsAbsorp( Atemp : real ): real;
var
    i : integer;
    Btemp : real;
    found : boolean;
begin
    if Atemp <= 0.300 then Atemp:=0.300;
    if Atemp >= 0.873 then Atemp:=0.873;
    found:=false;
    i:= 1;
    repeat
        if Atemp <= WL1[i] then
            begin
                Btemp:= Kv1[i-1]-(Kv1[i-1]-kv1[i])/(WL1[i]-WL1[i-1])
                    *(Atemp-WL1[i-1]);
                found:= true;
            end
            else
                i:= i + 1 ;
        until ( i >= 24 ) or ( found );
    GaAsAbsorp:= 4.0 * 3.141596 * Btemp / Atemp ;
end;

(* Si absorption coefficients vs. wavelength *)

Function SiAbsorp( Atemp : real ): real;
var
    i : integer;
    Btemp : real;
    found : boolean;
begin

```

```

if Atemp <= 0.294 then Atemp:=0.294;
if Atemp >= 1.125 then Atemp:=1.125;
found:=false;
i:= 1;
repeat
  if Atemp <= WL2[i] then
    begin
      Btemp:= Kv2[i-1]-(Kv2[i-1]-kv2[i])/(WL2[i]-WL2[i-1])
        *(Atemp-WL2[i-1]);
      found:= true;
    end
    else
      i:= i + 1 ;
until ( i >= 36 ) or ( found );
SiAbsorp:= 4.0 * 3.141596 * Btemp / Atemp ;
end;

```

(* Ge absorption coefficients vs. wavelength *)

```

Function GeAbsorp( Atemp : real ): real;
var
  i : integer;
  Btemp : real;
  found : boolean;
begin
  if Atemp <= 0.500 then Atemp:=0.500;
  found:=false;
  i:= 1;
  repeat
    if Atemp <= WL3[i] then
      begin
        Btemp:= Kv3[i-1]-(Kv3[i-1]-kv3[i])/(WL3[i]-WL3[i-1])
          *(Atemp-WL3[i-1]);
        found:= true;
      end
      else
        i:= i + 1 ;
until ( i >= 12 ) or ( found );
GeAbsorp:= 4.0 * 3.141596 * Btemp / Atemp ;
end;

```

```

Function AlGaAsAbsp( x, yw : real):real;
var
  i1, i2 : integer;
  flag1, flag2 : boolean;
  a1, a2, a3, y: real;

```

```

begin
  if x <= 0.099 then x:= 0.099 else
  if x >= 0.804 then x:= 0.804;
  y:= 1.24 / yw ;
  if y <= 1.500 then y:= 1.500 else
  if y >= 3.500 then y:= 3.500;
  Flag1:= false;
  i1:= 1;
  while not flag1 do
  begin
    if (x >= Fxx[i1]) and ( x <= Fxx[i1+1]) then
      Flag1:= true   else
      I1:= i1 + 1;
  end;
  Flag2:= false;
  i2:= 1;
  while not Flag2 do
  begin
    if ( y >= Fyy[i2]) and ( y <= Fyy[i2+1]) then
      Flag2:= true   else
      I2:= I2 + 1;
  end;
  a1:= Fxy[(i1-1) * 21 + i2] + (y - Fyy[I2]) / (Fyy[I2+1] - Fyy[I2])
    * (Fxy[(i1-1)*21 + i2 + 1] - Fxy[(i1-1)*21 + i2]);
  a2:= Fxy[i1 * 21 + i2 ] + (y - Fyy[i2]) / (Fyy[I2+1] - Fyy[i2])
    * (Fxy[i1*21 + i2 + 1] - Fxy[i1*21 + i2]);
  a3:= a1 - (a1 - a2) * (x - Fxx[i1]) / (Fxx[i1+1] - Fxx[i1]);
  AlGaAsAbsp:= a3;
end;

Function GaAsAlU(Tx, xx, Nd1, Na1:real; ii : integer): real;
begin
  if ii = 1 then
    GaAsAlU:= exp(ln10 * ( -1.5545 + 0.0106*xx + (0.735 + 0.0013*xx)*
      ln(Na1)/ln10 - (0.0253 + 0.0052*xx)*sqrt(ln(Na1)/ln10))* 300 / Tx) else
    GaAsAlU:= exp(ln10 * ( -9.723 + 0.0095*xx + (1.576 + 0.0012*xx)* ln(Nd1)
      /ln10 - (0.0507 + 0.0034*xx)*sqrt(ln(Nd1)/ln10))*exp(0.75*ln(300/Tx)));
end;

Function GaAsAlL(Tx, xx, Nd1, Na1:real; ii : integer):real;
var
  Uh1, Ue1,
  Uh0, Ue0 : real;
begin

```



```

if ii = 1 then
begin
  Ue1:= GaAsAlU(Tx, xx, Nd1, Na1, ii);
  Ue0:= GaAsAlU(Tx, 0.0, Nd1, Na1, ii);
  GaAsALL:= sqrt(Ue1/Ue0)*exp(-9.72*xx)*( - 210.06 + 27.254*ln(na1)/
    ln10 - 0.850*sqr(ln(Na1)/ln10))*exp(0.87*ln(Tx/300));
end else
begin
  Uh1:= GaAsAlU(Tx, xx, Nd1, Na1, ii);
  Uh0:= GaAsAlU(Tx, 0.0, Nd1, Na1, ii);
  GaAsALL:= sqrt(Uh1/Uh0)*exp(-9.72*xx)*( - 116.92 + 14.466*ln(Nd1)/
    ln10 - 0.438*sqr(ln(Nd1)/ln10))* Tx / 300 ;
end;
end;

```

```

(* (InGa)As absorption coefficients vs. wavelength *)
(* In --> 53 % ; Ga --> 47 % *)

```

```

Function InGaAsAbsorp(Atemp : real):real;
var
  Btemp : real;
begin
  if Atemp <= 0.74 then
    Btemp:= 10.0 else
  if Atemp <= 1.40 then
    Btemp:= exp((-1.51515*Atemp+6.12121 )*ln10) / 1E4 else
  if Atemp <= 1.50 then
    Btemp:= exp((-1.5491*Atemp+6.16874)*ln10) / 1E4 else
  if Atemp <= 1.54 then
    Btemp:= exp((-3.65495*Atemp+9.32752)*ln10) / 1E4 else
  if Atemp <= 1.73 then
    Btemp:= exp((-14.2051*Atemp+25.5749)*ln10) / 1E4 else
    Btemp:= 1E-3;
  InGaAsAbsorp:= Btemp;
end;

```

```

(* Absorption Coefficients Calculations of Each Cell*)

```

```

Function Absp(Name : CellType; Wlength: real ): real;
var
  Atemp : real;
begin
  Atemp:= Wlength;
  if Name = GAAS then

```

```

    Absp:=GaAsAbsorp(Atemp)      else
    if Name = ALGAAS then
    Absp:=AlGaAsAbsp(alratio, Atemp)  else
    if Name = INGAAS then
    Absp := INGAASAbsorp(Atemp) else
    if Name = GE      then
    Absp:=GeAbsorp(Atemp)      else
    if Name = SI      then
    Absp:=SiAbsorp(Atemp);
end;

(* to interpolate the irradiance power density of AM? *)

function AMX( x : real; ii : integer):real;
var
    i , j : integer;
    xx : real ;
    found : boolean;
begin
    Found:=false;
    i:=1;
    while ( not Found) and ( i <=195 ) do
    begin
        if ( X = Atype[i] ) then
            begin
                Found:=true;
                AMX := Btype[ii,i] * 1E3;
            end
            else
                if ( x < Atype[i] ) and ( x > Atype[i+1] ) then
                    begin
                        XX:=(x - Atype[i+1]) * (Btype[ii,i+1] - Btype[ii,i]) /
                            (Atype[i+1] - Atype[i]) + Btype[ii,i+1];
                        AMX:=XX * 1E3;
                        Found:=true;
                    end
                    else
                        i:=i+1;
                    end;
            end;
    end;

function cosh( x : real ) : real;
begin
    if (x) >= 87.0 then      (* to avoid out of range of exp(x) *)
        x:= 87.0      ELSE
    if x <= - 87.0 then

```

```

        x:= -87;
        cosh:=( exp(x) + exp(-x) ) / 2.0;
end;

function sinh( x : real ) : real;
begin
    if x >= 87.0 then          (* same as that of function cosh *)
        x:= 87.0   else
    if x <= -87.0 then
        x:= -87.0;
    sinh:=( exp(x) - exp(-x) ) / 2.0;
end;

(* Calculate the Incident Photon Number vs. sunlight wavelength *)
(* X1 --> wavelenth ; X2 --> irradiance power density.          *)

Function NphCal(X1,X2:real):real;
begin
    NphCal:= 4.06E18 *X1 *sqr(X1) * X2 * 1E-3 ;
end;

Function QF( ii, Tx : integer; Wlength: real ): real;
var
    i, j : integer;
    C1, C2,
    D1, D2,
    A1, A2,
    A3, Flux : real;
begin
    i:= ii;
    Flux:= NphCal(Wlength,AMX(Wlength,AirM));
    if not GFactor then
        QF:= Qe * Flux   else
    begin
        if Tx = 1 then    (* default AlGaAs,GaAs,InGaAs,Si or Ge system *)
            QF:= Qe * Flux   else
        if Tx = 2 then
            begin
                A1:= Absp(CellName[i-1],Wlength );
                D1:= Tj[i-1];
                C1:= A1 * D1 ;
                if C1 >= 87.0 then    (* to avoid out of range *)
                    C1:= 87.0;
                QF:= Qe * Flux * exp(-C1);
            end
        end
    end
end;

```

```

end    else
if Tx = 3 then
begin
    A1:= Absp(cellName[i-2] ,Wlength);
    D1:= Tj[i-2];
    A2:= Absp(CellName[i-1],Wlength);
    D2:= Tj[i-1];
    C1:= A1 * D1 ;
    C2:= A2 * D2 ;
    if C1 >= 87.0 then
        C1:= 87.0;
    if C2 >= 87.0 then
        C2:= 87.0;
    QF:= Qe * Flux * exp(-C1 ) * exp(-C2);
end;
end;
end;

(* Calculate short-circuit current density of each cell *)

Function Jpnd(ii, jj, Tx : integer; Name : CellType; Wlength : real ):real;
var
    X0, X1, XS, X2, X3, X4, X5, X6,
    Jx, X7, XQF, Absorb, C1, C2 : real;
    i, j : integer;
begin
    i:= ii ;
    j:= jj ;
    XQF:= QF(i,Tx, Wlength) ;
    if GFactor then
    begin
        Absorb:= Absp(Name, Wlength);
        if j = 1 then
        begin
            if PN then
            begin
                X0:= Lnn[i] ;
                X1:= Tn[i] ;
                XS:= Sn[i];
            end else
            begin
                X0:= Lpp[i];
                X1:= Tp[i];
                XS:= Sp[i];
            end
        end
    end
end

```

```

end;
X2:= Absorb * X0 ;
X3:= XS * 1E4 * X1 / X0 ;
X4:= cosh(Xj[i]/X0);
X5:= sinh(Xj[i]/X0);
X7:= EXP(-Absorb*Xj[i]);
Jx := XQF*X2/(sqr(X2)-1)*((X2+X3*(1-X7*X4)-X7*X5
)/ ( X3*X5 + X4) - X2*X7);
end else
if j = 2 then
begin
    if PN then
    begin
X0:= Lpp[i] ;
X1:= Tp[i] ;
XS:= Sp[i] ;
        end else
        begin
X0:= Lnn[i];
X1:= Tn[i];
XS:= Sn[i];
        end;
X2:= Absorb * X0 ;
X3:= XS * 1E4 * X1 / X0 ;
X4:= cosh((Tj[i] - Xj[i] - Wj[i])/X0);
X5:= sinh((Tj[i] - Xj[i] - Wj[i])/X0);
C1:= absorb * (Tj[i] - Xj[i] - Wj[i]);
C2:= absorb * (Xj[i] + Wj[i]);
if C1 >= 87.0 then
    C1:= 87.0;
if C2 >= 87.0 then
    C2:= 87.0;
X6:= exp(-C1);
X7:= XQF*X2/(sqr(X2)-1)*EXP(-C2);
(* check overflow *)
if ((ln(abs(x3))/ln10 + ln(abs(x4))/ln10) >= 37.0 ) or
    ((ln(abs(x3))/ln10 + ln(abs(x5))/ln10) >= 37.0 ) then
begin
x4:= x4 / 1E10;
x5:= x5 / 1E10;
x6:= x6 / 1E10;
        end ;
Jx :=X7 * (X2 - (X3 * (X4 - X6) + X5 + X2 * X6)
/ (x3 * X5 + X4 ) );

```

```

end   else
begin
    C1:= Absorb * Xj[i];
    C2:= Absorb * Wj[i];
    if C1 >= 87.0 then
        C1:= 87.0 ;
    if C2 >= 87.0 then
        C2:= 87.0;
    Jx  := XQF*exp(-C1)*(1-exp(-C2));
end;
end   else
Jx  := XQF;
Jpnd:= Jx;
end;

Function TransM(ii : integer; Name: CellType; Upper, Lower : real): real;
var
    csub, C1,
    cnew, cold,
    X, X0, X1,
    X2, X3, H : real;
    i          : integer;
    cfound : boolean;
    citer : integer;
begin
    citer:= 5;
    cold:= 0.0;
    cfound:= false;
    repeat
        H:= (Upper - Lower ) / ( 2.0 * citer );
        cnew:= 0.0 ;
        X1:=0.0; X2:=0.0;
        c1:= absp(Name,1.24/lower) * Tj[ii];
        if c1 >= 87 then
            c1:= 87;
        x0:= exp(- C1);
        c1:= absp(Name,1.24/upper) * Tj[ii];
        if c1 >= 87 then
            c1:= 87;
        x0:= x0 + exp(- C1) ;
        for i:=1 to ( 2 * citer - 1) do
            begin
                X := 1.24 / (Lower + i * H);
                c1:=absp(Name, x) * Tj[ii];

```

```

    if c1 >= 87 then
        c1:= 87;
    x3:= exp(- C1);
    if odd(i) then
        X1:= X1 + X3   else
        X2:= X2 + X3   ;
end;
csub:= H * ( X0 + 2.0 * X2 + 4 * X1 ) / 3.0 ;
cnew:= csub;
If (abs(cnew - cold) < 0.0001) or (citer >= 100) then
begin
    CFound:= true ;
    if citer >= 100 then
        writeln(' ---> Exceed 100 iteration in CollEff');
    end else
    begin
        citer:= citer + 1;
        cold:= cnew;
    end;
Until ( cfound);
TransM := cnew;
end;

```

(* use simpson composite method to integrate the Jsc *)

```

Function  JscTot(ii, Tx: integer; Name: CellType; Upper, Lower : real): real;
var
    Jnew, Jold,
    X, X0, X1,
    X2, X3, H : real;
    Jsub      : array [1..3] of real;
    i, j, k : integer;
    Jfound : boolean;
    Jiter : integer;
begin
    Jiter:= 5;
    Jold:= 0.0;
    Jfound:= false;
    repeat
        H:= (Upper - Lower ) / ( 2.0 * Jiter );
        if not GFactor then
            K:= 1           else
            K:= 3;
        Jnew:= 0.0 ;
    
```

```

for j:= 1 to k do
begin
  X1:=0.0; X2:=0.0;
  x0:= Jpnd(ii,j,Tx,Name,1.24/Lower ) + Jpnd(ii,j,Tx,Name,1.24/Upper);
  for i:=1 to ( 2 * Jiter - 1) do
  begin
    X := 1.24 / (Lower + i * H);
    X3:=Jpnd(ii,j,Tx,Name,X);
    if odd(i) then
      X1:= X1 + X3   else
      X2:= X2 + X3   ;
  end;
  Jsub[j]:= H * ( X0 + 2.0 * X2 + 4 * X1 ) / 3.0 ;
  Jnew:= Jnew + Jsub[j];
end;
If (abs(Jnew - Jold) < 0.0001) or (Jiter >= 100)  then
begin
  JFound:= true ;
  if Jiter >= 100 then
    writeln(' ---> Exceed 100 iteration in Jsc ');
  end else
  begin
    Jiter:= Jiter + 1;
    Jold:= Jnew;
  end;
  Until ( Jfound);
  JscTot:= Jnew;
end;

```

(* use Binary search method to obtain the Maximum voltage, Vm *)

```

Procedure GetVm(Xsc,Xoo : real; k : integer);
var
  i : integer;
  Diff, XVt,
  x, X1, x2: real;
  Found : boolean;
begin
  Found:=false;
  x1:= 0.1;
  x2:= Eg[k];
  REPEAT
    x:= (x1 + x2) / 2.0 ;
    XVt:= x / (IdealF[k]* 0.0259 * Tratio) ;

```



```

Diff:= Ln((1 + Xsc/Xoo)/(1 + Xvt) ) - Xvt ;
if (ABS(Diff / xvt) <= 0.001) then
  Found:=true  else
  if Diff >= 0.0  then
    x1:= x      else
    x2:= x ;
until (Found) or ( x1 > x2);
if Found=true  then
  Vm[k] :=x      else
  Vm[k] :=0.0;
end;

Procedure GetAda( x : real; k:integer) ;
begin
  if RGmodel[k] = DIFFUSION  then
    Joo[k]:= Qe * sqr(Ni[k]) * (Dpp[k] / (Lpp[k] * Ndd[k])
    + Dnn[k] / (Lnn[k] * Na[k]))  else
    Joo[k]:= Qe * Wj[k] * 1E-4 * Ni[k] / sqrt(Tn[k] * Tp[k]);
  Voc[k] := IdealF[k] * 0.0259 * Tratio * ln( Jsc/Joo[k]+ 1 );
  GetVm(Jsc,Joo[k],k);
  FF[k]:=Vm[k]/Voc[k]*(1.0-(exp(Vm[k]/(IdealF[k]*0.0259 * Tratio))-1)
  /(exp(Voc[k]/(IdealF[k]*0.0259 * Tratio)) - 1) );
  Ada[k]:= Jsc * Voc[k] * FF[k] / (Pam * 1E-3) ;
end;

Procedure SingleJunction;
begin
  Low:= Eg[1];
  Jsc:= JscToT(1,1,CellName[1],high,low);
  Coleff[1]:= 1 - TransM(1,CellName[1],high,low);
  Jsc:= Jsc * Coleff[1];
  GetAda(Eg[1],1);
  AdaTot:= Ada[1];
end; (* of SingleJunction *)

procedure Doublejunction;
var
  k, II,
  i, j : integer;
  found: boolean;
  x      : real;
  Jtop,
  Jbot1, Jbot2,
  Jbot : real;

```

```

begin (* of Double *)
  Jtop:= JscToT(1,1,CellName[1],high,Eg[1]);
  Coleff[1]:= 1 - TransM(1,CellName[1],high,Eg[1]);
  Jtop:= Jtop * Coleff[1];
  Jbot1:= JscToT(2,1,CellName[2],Eg[1],Eg[2]);
  Coleff[2]:= 1 - TransM(1,CellName[2],Eg[1],Eg[2]);
  Jbot1:= Jbot1 * Coleff[2];
  Jbot2:= JscToT(2,2,CellName[2],High,Eg[1]);
  Jbot:= Jbot1 + Jbot2;
  if Jtop >= Jbot then
    Jsc:= Jbot    else
    Jsc:= Jtop;
  Getada(Eg[1],1);
  GetAda(Eg[2],2);
  AdaToT:= Ada[1] + Ada[2];
end;

```

```

procedure triplejunction;

```

```

var

```

```

  i, j, k,
  m, n, o : integer;
  x1, x2  : real;
  Jtop,
  Jmid,
  Jbot1,
  Jbot2,
  Jbot3,
  Jbot    : real;
  found   : boolean;

```

```

Procedure AtripleAda;

```

```

begin

```

```

  if (Jtop <= Jmid) and (Jtop <= Jbot) then
    Jsc:= Jtop    else
    if (Jtop >= Jmid) and (Jtop <= Jbot) then
      Jsc:= Jmid    else
      if (Jtop <= Jmid) and (Jtop >= Jbot) then
        Jsc:= Jbot    else
        if (Jtop >= Jmid) and (Jtop >= Jbot) then
          begin
            if Jmid >= Jbot then
              Jsc:= Jbot    else
              Jsc:= Jmid;

```

```

    end;
    GetAda(Eg[1],1);
    GetAda(Eg[2],2);
    GetAda(Eg[3],3);
    AdaToT:= Ada[1] + Ada[2] + Ada[3];
end; (* of AtripleAda *)

begin (* triplejunction *)
    x2:= Eg[1];
    x1:= Eg[2];
    Low:= Eg[3];
    Jtop:= JscToT(1,1, CellName[1],High,x2);
    Coleff[1]:= 1 - TransM(1,CellName[1],high,x2);
    Jtop:= Jtop * Coleff[1];
    Jbot1:= JscToT(2,1,CellName[2],x2,x1);
    Coleff[2]:= 1 - TransM(2,CellName[2],x2,x1);
    Jbot1:= Jbot1 * Coleff[2];
    Jbot2:= JscToT(2,2,CellName[2],High,x2);
    Jmid:= Jbot1 + Jbot2;
    Jbot1:= JscToT(3,1,CellName[3],x1,low);
    Coleff[3]:= 1 - TransM(3,CellName[3],x1,low);
    Jbot1:= Jbot1 * Coleff[3];
    (* Jbot2:= JscToT(2,2,CellName[2],x2,x1);
    Jbot3:= JscToT(1,3,CellName[1],high,x1);
    Jbot:= Jbot1 + Jbot2 + Jbot3;
    *)
    Jbot:= jbot1;
    ATripleAda;
end;

procedure GetNi(Name:CellType; i:integer);
var
    lh, hh,
    x1, x2 : real;
begin
    if Name = SI then
    begin
        Mn[i]:= 1.1 ;
        Mp[i]:= 0.56;
        Eg[i]:= Bandgap(Name);
    end else
    if Name = GAAS then
    begin
        Mn[i]:= 0.067;

```

```

    Mp[i]:= 0.641;
    Eg[i]:= Bandgap(name);
    Eg[i]:= Eg[i] - 1.6E-8 * EXP(1/3 * LN(NA[I]) );
end   else
if Name = ALGAAS then
begin
    Mn[i]:= 0.067 + 0.083 * alratio;
    lh:= 0.087 + 0.063 * alratio;
    hh:= 0.62 + 0.14 * alratio;
    Mp[i]:= exp(2.0/3.0*ln(lh*sqrt(lh) + hh*sqrt(hh)));
    Eg[i]:= bandgap(name);
end   else
If Name = GE      then
begin
    Mn[i]:= 0.55 ;
    Mp[i]:= 0.37 ;
    Eg[i]:= bandgap(name);
end   else
begin
    Mn[i]:= 0.041;
    Mp[i]:= 0.5;
    Eg[i]:= bandgap(name);
end;
x1:= Mn[i] * Mp[i];
Ni[i]:= sqrt(x1 * sqrt(x1) * 2.33E31) * Tdegree * sqrt(Tdegree)
        * exp(-Eg[i] / (0.0259 * 2 * Tratio));
end;(* of getNi *)

Procedure GetImplicit(ii:integer);
var
    Eps,
    xratio,
    Vbi : real;
    i : integer;
begin
    if CellName[ii] = SI then
    begin
        Eps:= 12.0;
        Mobe[ii]:= 88.0 * exp(-0.57*ln(Tratio)) + 7.4E8 * exp(-2.33*ln(Tdegree))
            /(1+0.88*exp(-0.146*ln(Tratio))*(Ndd[ii]/(1.26E17*exp(2.4*ln(Tratio)))));
        Mobp[ii]:= 54.3 * exp(-0.57*ln(Tratio)) + 1.36E8 * exp(-2.23*ln(Tdegree))
            /(1+0.88*exp(-0.146*ln(Tratio))*(Na[ii]/(2.35E17*exp(2.4*ln(Tratio)))));
        Dnn[ii]:= 0.0259 * Tratio * Mobe[ii];
        Dpp[ii]:= 0.0259 * Tratio * Mobp[ii];
    end;
end;

```

```

Tn[ii]:= 12E-6 / (1 + Na[ii] / 5E16);
Tp[ii]:= 12E-6 / (1 + Ndd[ii] / 5E16);
Lnn[ii]:= sqrt(Tn[ii]*Dnn[ii]) * 1E4 ; (* um *)
Lpp[ii]:= sqrt(Tp[ii]*Dpp[ii]) * 1E4 ;
end   else
if (CellName[ii] = GAAS) or (CellName[ii] = ALGAAS) then
begin
if CellName[ii] = GAAS then
begin
Xratio:= 0.0;
Eps:= 12.9;
end   else
begin
if Eg[ii] <= 1.99 then
Alratio:= (Eg[ii] - 1.424) / 1.247;
Xratio:= Alratio;
Eps:= 12.9 *(1 - Alratio) + 10.06 * alratio;
end;
Mobe[ii]:= GaAsALU(Tdegree,xratio ,Ndd[ii],Na[ii],1);
Mobp[ii]:= GaAsALU(Tdegree,xratio ,Ndd[ii],Na[ii],2);
Dnn[ii]:= 0.0259 * Tratio * Mobe[ii];
Dpp[ii]:= 0.0259 * Tratio * Mobp[ii];
Lnn[ii]:= GaAsALL(Tdegree,xratio,Ndd[ii],Na[ii],1);
Lpp[ii]:= GaAsALL(Tdegree,xratio,Ndd[ii],Na[ii],2);
Tn[ii]:= sqrt(Lnn[ii]) / Dnn[ii] * 1E-8 ;
Tp[ii]:= sqrt(Lpp[ii]) / Dpp[ii] * 1E-8 ;
end   else
if (CellName[ii] = GE ) then
begin
Mobe[ii]:= 695.0;
Mobp[ii]:= 540.0;
Dnn[ii]:= 18.0;
Dpp[ii]:= 14.0;
Lnn[ii]:= 232.0;
Lpp[ii]:= 100.0;
Tn[ii]:= 3E-5;
Tp[ii]:= 7.2E-6;
Ndd[ii]:= 2E19;
Na[ii]:= 2E17;
Eps:= 16.0;
end   else
begin
Eps:= 12.0;
Mobe[ii]:= 6000.0;

```

```

    Mobp[ii]:= 180.0;
    dnn[ii]:= 155.4;
    Dpp[ii]:= 4.66;
    Lnn[ii]:= 6.0;
    Lpp[ii]:= 2.0;
    Tn[ii]:= 2.3E-9;
    Tp[ii]:= 8.59E-9;
    Ndd[ii]:= 5E17;
    Na[ii]:= 2E17;
end;
GetNi(CellName[ii],ii);
Vbi:= 0.0259 * Tratio * ln(Na[ii] / sqr(Ni[ii]) * Ndd[ii] );
Wj[ii]:= sqrt(2 * Eps * Pemit * Vbi / Qe *(1/Na[ii] + 1/Ndd[ii])) *1E4 ;
end;

Function Ranu(var lo, hi :real):real;Alien;

Procedure InitialGuess;
var
    i, j : integer;
    ll,hh: real;
begin
    ll:= 0.0;
    hh:= 1.0;
    for i:= 1 to (IndVar+1) do
        for j:= 1 to IndVar do
            InitGuess[i,j]:= Lbound[j] + (Ubound[j] - lbound[j])*Ranu(ll,hh);
        end;
    end;

Procedure Radiation(II:integer);
begin
    Lnn[ii]:= sqrt(1/(1/sqr(Lnn[ii]*1E-4) + Kln[ii] * fluxx)) * 1E4;
    Lpp[ii]:= sqrt(1/(1/sqr(Lpp[ii]*1E-4) + Klp[ii] * fluxx)) * 1E4;
    Tn[ii]:= 1 / (1/Tn[ii] + Dnn[ii] * Kln[ii] * fluxx);
    Tp[ii]:= 1 / (1/Tp[ii] + Dpp[ii] * Klp[ii] * fluxx);
end;

Function GetF(ii:integer):real;
var
    i, j : integer;
begin
    i:= ii;
    repeat
        Ndd[i]:=exp(Guess[(i-1)*5+1] * ln10);
    until

```

```

Na [i]:=exp(guess[(i-1)*5+2] * ln10);
Xj[i]:= Guess[(i-1)*5 + 3];
Tj[i]:= Guess[(i-1)*5 + 4];
if ( i = 1 ) then
    Eg[1]:= Guess[(i-1)*5 + 5]   else
if i = 2      then
    Eg[2]:= Guess[(i-1)*5 + 5]   else
if (i = 3)   then
    Eg[3]:= Guess[(i-1)*5 + 5];
GetImplicit(i);
if particle <> 0 then
    Radiation(i);
i:= i - 1;
until ( i <= 0 );
if ii = 1 then
    SingleJunction   else
if ii = 2 then
    Doublejunction   else
    Triplejunction;
RealAda;
if Insolation > 1 then
    HighInsolation;
GetF:= AdaTot;
end;(* of GetF *)

```

```

Procedure FBestWorst(Var best, worst:integer);
var

```

```

    i, j : integer;
    Temp : real;
    found: boolean;
begin
    temp:= AdaVec[i];
    found:= false;
    for i:= 2 to (IndVar+1) do
    begin
        if (AdaVec[i] > temp ) then
            begin
                temp:= AdaVec[i];
                best:= i;
                found:= true;
            end;
        end;
    end;
    if not found then
        best:= 1 ;
    end;

```

```

temp:= AdaVec[1];
found:= false;
for i:= 2 to (IndVar+1) do
begin
  if (AdaVec[i] < temp) then
  begin
    Temp:= AdaVec[i];
    worst:= i;
    found:= true;
  end;
end;
if not found then
worst:= 1;
Fworst:= temp;
end;

Procedure Deworst(Var best, worst:integer);
var
  i, j : integer;
  a1,ll,hh,
  Sum : real;
begin
  ll:=0.0; hh:=1.0;
  For i:= 1 to IndVar do
    PreGuess[i]:= InitGuess[worst,i];
  for i:= 1 to IndVar do
  begin
    Sum:= 0.0;
    for J:= 1 to (IndVar+1) do
      sum:= sum + InitGuess[j,i] ;
    AveX[i]:=(Sum - PreGuess[i]) / IndVar;
    InitGuess[worst,i]:= 1.3 *(AveX[i] - PreGuess[i]) + AveX[i];
    Guess[i]:= InitGuess[worst,i];
  end;
end;

Procedure notBetter(var be, wo : integer);
var
  i, j : integer;
begin
  for I:= 1 to IndVar do
  begin
    Guess[i]:= (Guess[i] + InitGuess[be,i]) * 0.5;
    InitGuess[wo,i]:= Guess[i];
  end;
end;

```



```

    end;
end;

Procedure CheckBound;
var
    i : integer;
begin
    for i:= 1 to IndVar do
        begin
            if Guess[i] >= Ubound[i] then
                Guess[i]:= Ubound[i] else
            if Guess[i] <= Lbound[i] then
                Guess[i]:= Lbound[i];
        end;
    end;
end;

Procedure Printout;
var
    i, j : integer;

begin (* main of Printout *)
    For i:= 1 to CellConf do
        begin
            writeln(out, 'Cell # ',I:2,' ----> ');
            writeln(out, ' Na : ',Na[i]);
            writeln(out, ' Nd : ',Ndd[i]);
            writeln(out, ' Xj : ',Xj[i]);
            writeln(out, ' Tj : ',Tj[i]);
            Writeln(out, ' Sp : ',Sp[i]);
            writeln(out, ' Sn : ',Sn[i]);
            writeln(out, ' Voc: ',Voc[i]);
            writeln(out, ' F.F. ',FF[i]);
            writeln(out, ' Ada: ',Ada[i]);
            WRITELN(OUT, ' Ln : ',Lnn[i]);
            writeln(out, ' Lp : ',Lpp[i]);
            writeln(out, ' Tn : ',Tn[i]);
            writeln(out, ' Tp : ',Tp[i]);
            writeln(out, ' Mobe:',mobe[i]);
            writeln(out, ' Mobp:',mobp[i]);
            writeln(out, ' Ni : ',Ni[i]);
            writeln(out, ' Wj : ',Wj[i]);
            writeln(out, ' Mn : ',Mn[i]);
            writeln(out, ' Mp : ',Mp[i]);
            writeln(out, ' Eg : ',Eg[i]);
        end;
    end;
end;

```

```

    writeln(out, ' Col: ', ColEff[i]);
end;
writeln(out, ' Jsc : ', jsc);
Writeln(out, ' ToTal Ada : ', AdaTot);
writeln(out, ' Air Mass : ', Airmass);
Writeln(out, ' Sun Insolation : ', insolation);
writeln(out, ' Temperature : ', Tdegree);
if Btunnel then
  for i:= 1 to TunnelNum do
    writeln(OUT, ' Tunnel drop : ', TdropV[i]);
end; (* Printout*)

Procedure Box(ii:integer);
var
  index, count,
  Be,Wo,
  i, j : integer;
  Spt,
  found: boolean;
begin
  InitialGuess;
  For i:= 1 to (IndVar+1) do
    begin
      for j:= 1 to IndVar do
        Guess[j]:= InitGuess[i,j];
        AdaVec[i]:= GetF(ii);
      end;
      FbestWorst(Be,Wo);
      index:= wo;
      Deworst(Be,Wo);
      Checkbound;
      AdaVec[Wo]:= GetF(ii);
      Found:= FALSE;
      repeat
        if (AdaVec[Wo] <= Fworst) then
          begin
            NotBetter(be,wo) ;
            Checkbound;
            AdaVec[Wo]:= GetF(ii);
          end
        else
          begin
            FBestWorst(Be,Wo);
            if (ABS((AdaVec[Wo]-AdaVec[Be])/AdaVec[Be]) <= error) then
              Found:= true
            else

```

```

        begin
        if wo = index then
            Notbetter(be,wo) else
        begin
            DeWorst(Be,Wo);
            Index:= wo;
        end;
        Checkbound;
        AdaVec[Wo]:= GetF(ii);
        end;
    end;
until (Found);
end;

begin
    Initial;
    ReadInput;
    if system then
        PruneSearch else
    begin
        CellSelection;
        InputBound;
    end;
    ARcoating;
    Contact;
    TunnelSelection;
    Irradiation;
    writeln(' The highest Cutoff Eg. < 4.0. eV ');
    readln(High);
    writeln(' Relative Error for convergence detections. <= 1E-4 ');
    readln(error);
    Step:= 1;
    Fbest:= 0.15;
    writeln(' Maximum Iteration? ');
    readln(Iter);
repeat
    Box(CellConf);
    printout;
    Step:=Step + 1;
    if (AdaToT >= Fbest) and (AdaToT <= 0.35) then
        Fbest:= AdaToT;
Until (Step >= iter) or (AdaTot > 0.35 );
    ARthickness;
    writeln(out,' Shadow : ',shadow);

```

```
writeln(out,' Reflection : ',RefL);  
writeln(out,' LossRatio : ',LossRatio);  
writeln(out,' Series Resistance : ',Rs);  
end.
```

APPENDIX D
INPUT PARAMETERS FOR THE OPTIMAL DESIGN OF
(ALGA)AS/GAAS/IN_{0.53}GA_{0.47}AS THREE-JUNCTION SOLAR CELLS

Table D.1 Physical parameters for the In_{0.53}Ga_{0.47}As bottom cell.

Electron lifetime τ_n (s)	2.3×10^{-9}
Hole lifetime τ_p	8.58×10^{-9}
Electron diffusion length L_n (μm)	6.0
Hole diffusion length L_p	2.0
P-dopant density N_A (cm^{-3})	2×10^{17}
N-dopant density N_D	5×10^{17}
Electron mobility μ_n ($\text{cm}^2/\text{V-s}$)	6000
Hole mobility μ_p	180
Electron effective mass $m_n(m_0)$	0.041
Hole effective mass m_p	0.5
Bandgap Energy E_g (eV) at 300 K	0.75

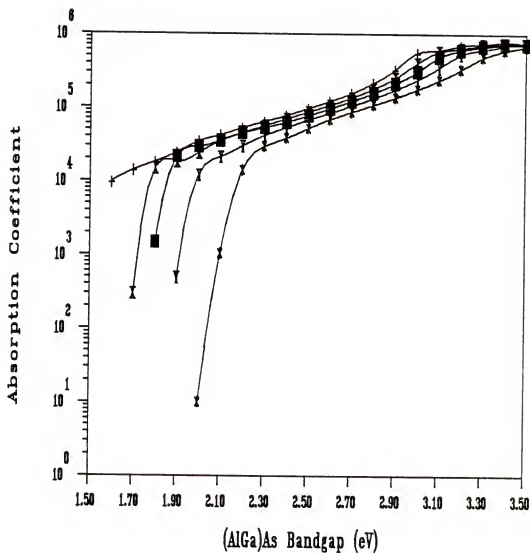


Fig. D.1 Absorption coefficients for various Al composition of (AlGa)As top cell. + : 14 %, X : 22 %, solid square : 30 %, Y : 38 % and * : 46%.

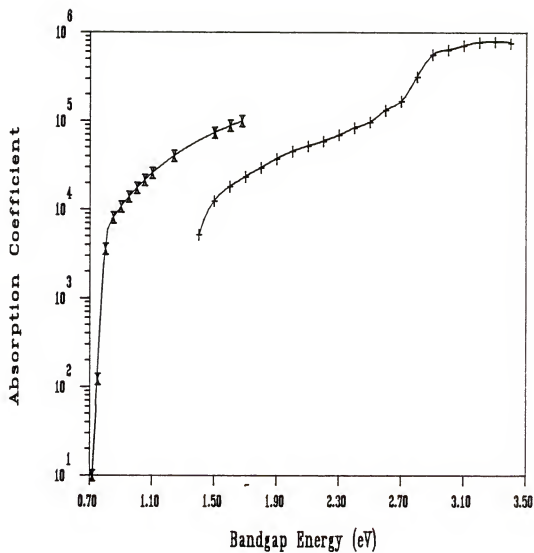


Fig. D.2 Absorption coefficients for GaAs and $\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$ solar cell materials. X : $\text{In}_{0.53}\text{Ga}_{0.47}\text{As}$ and + : GaAs .

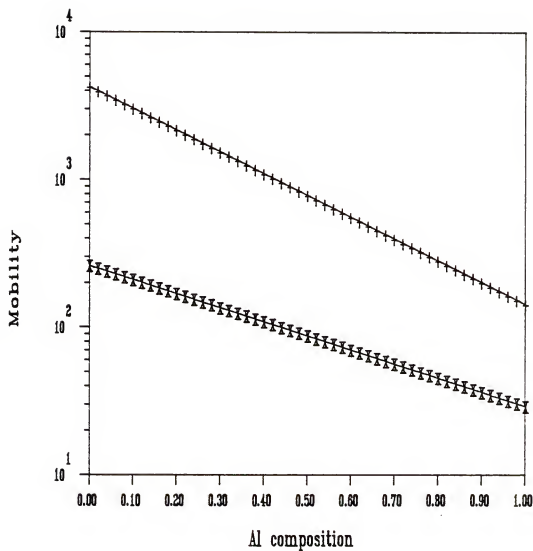


Fig. D.3 The Al composition dependence of minority carrier mobility for (AlGa)As top cell at room temperature and doping density of 10^{17} cm^{-3} . + : electron mobility and X : hole mobility.

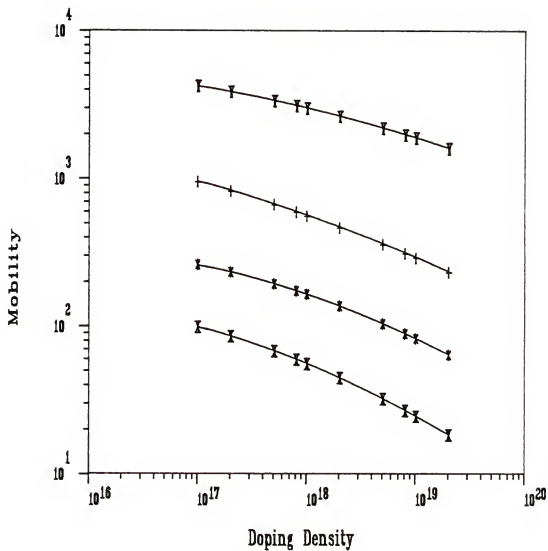


Fig. D.4 The doping density dependence of minority carrier mobility for $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}$ and GaAs. + : $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}$ electron mobility, X : $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}$ hole mobility, Y : GaAs electron mobility and * : GaAs hole mobility.

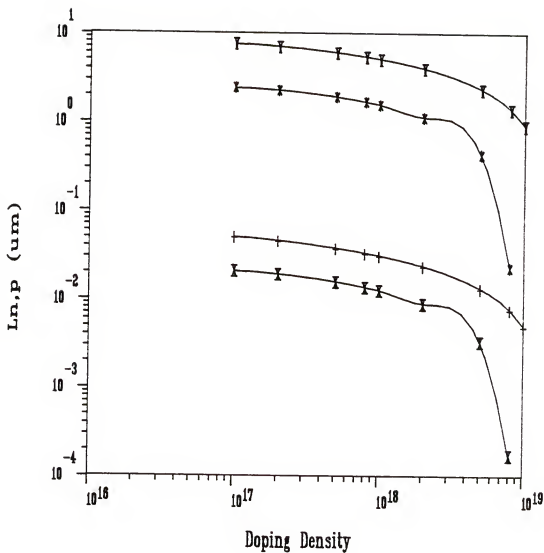


Fig. D.5 The doping density dependence of minority carrier diffusion length for $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}$ and GaAs. + : $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}$ electron diffusion length, X : $\text{Al}_{0.44}\text{Ga}_{0.56}\text{As}$ hole diffusion length, Y : GaAs electron diffusion length and * : GaAs hole diffusion length.

REFERENCES

- [1] S. M. Sze, Physics of Semiconductor Devices (Wiley, New York, 1981).
- [2] R. R. Ferber, E. N. Lostogue and K. Shimada, JPL Publication, 84-71(1984).
- [3] E. D. Jackson, Proceedings of Solar Energy Conference, 122(1955).
- [4] J. J. Loferski, J. Appl. Phys. 27, 777(1956).
- [5] S. M. Bedair, M. F. Lamorte and J. R. Hauser, Appl. Phys. Lett. 34, 38(1979).
- [6] M. F. Lamorte and D. Abbott, Solid State Electronics 22, 467(1979).
- [7] J. C. C. Fan, Proceedings of 16th IEEE Photovoltaic Specialists Conference, 142(1982).
- [8] C. Flores, Proceedings of 16th IEEE Photovoltaic Specialists Conference, 562(1982).
- [9] H. C. Hamaker, Proceedings of 17th IEEE Photovoltaic Specialists Conference, 140(1984).
- [10] P. K. Chiang, M. L. Timmons, G. G. Fountain and J. A. Hutchby, Proceedings of 18th IEEE Photovoltaic Specialists Conference, 562(1985).
- [11] J. A. Hutchby, R. J. Markunas, M. L. Timmons, P. K. Chiang and M. S. Bedair, Proceedings of 18th IEEE Photovoltaic Specialists Conference, 20(1985).
- [12] M. E. Nell and A. M. Barnetl, IEEE Trans. Electron Devices 34, 257(1987).
- [13] L. D. Partain, L. M. Fraas, P. S. Mcleod, J. A. Cape and M. S. Kuryla, J. Appl. Phys. 62, 694(1987).
- [14] J. L. Educato, M. Wagner and J. P. Leburton, J. Appl. Phys. 63, 540(1988).
- [15] L. M. Fraas and R. C. Knechtli, Proceedings of 13th IEEE Photovoltaic Specialists Conference, 886(1978).
- [16] C. R. Lewis, C. W. Ford, G. F. Virshup, B. A. Arau, R. T. Green and J. G. Werthen, Proceedings of 18th IEEE Photovoltaic Specialists Conference, 556(1985).

- [17] B. C. Chiang, H. C. Hamaker, G. E. Virshup and J. G. Werthen, Technical Digest of 3rd International Photovoltaic Science and Engineering Conference, 792(1987).
- [18] L. M. Fraas, B. K. Shin, J. A. Cape, R. A. Ransom and D. E. Sawyer, Proceedings of 16th Photovoltaic Specialists Conference, 655(1982).
- [19] R. C. Knechtli, R. Y. Loo and G. S. Kamath, IEEE Trans. Electron Devices 31, 577(1984).
- [20] H. J. Hovel, Solar Cells in Semiconductors and Semimetals (Academic Press, New York, 1975).
- [21] H. C. Hamaker, J. Appl. Phys. 58, 2344(1985).
- [22] H. J. Hovel and J. M. Woodall, Proceedings of 10th Photovoltaic Specialists Conference, 25(1973).
- [23] M. Yamaguchi and C. Amano, J. Appl. Phys. 57, 537(1985).
- [24] J. G. Werthen, H. C. Hamaker, G. F. Virshup and C. W. Ford, Appl. Phys. Lett. 46, 776(1985).
- [25] J. G. Werthen, G. F. Virshup and C. W. Ford, H. C. Hamaker, Appl. Phys. Lett. 48, 74(1986).
- [26] L. D. Partain, M. S. Kuryla, R. E. Weiss, R. A. Ransom, P. S. MacLeod, L. M. Fraas and J. A. Cape, J. Appl. Phys. 62, 3010(1987).
- [27] S. Yoshida, K. Mitsui, T. Oda, M. Kato, Y. Yukimoto and S. Matsuda, Proceedings of 17th IEEE Photovoltaic Specialists Conference, 42(1984).
- [28] M. Kato, K. Mitsue, K. Mizuguchi and K. Fujikawa, Proceedings of 18th IEEE Photovoltaic Specialists Conference, 14(1985).
- [29] M. J. Ludowise and W. T. Dietze, J. Appl. Phys. 55, 4318(1984).
- [30] Y. C. M. Yeh, K. I. Chiang and J. L. Tandon, Proceedings of 17th IEEE Photovoltaic Specialists Conference, 36(1984).
- [31] C. Amano, H. Sugiura and A. Yamamoto, Technical Digest of 3rd International Photovoltaic Science and Engineering Conference, 771(1987).
- [32] M. J. Box, The Computer Journal 8, 42(1966).
- [33] P. Harmon and David King, Expert Systems (John Wiley and Sons, New York, 1985).
- [34] A. Barr, P. R. Cohen and E. A. Feigenbaum, (Eds.), The Handbook of Artificial Intelligence (William Kaufman, Los Altos, 1981).

- [35] J. W. Wilson, J. J. Stith and L. V. Stock, NASA Technical Paper 2242 (1983).
- [36] J. Y. Yaung, NASA Conference Publication 2314, 56(1983).
- [37] N. L. Paterson and S. D. Harkness (Eds.), Proceedings of American Society for Metal Conference, 59(1976).
- [38] G. J. Dienes and G. H. Vineyard, Radiation Effects in Solids (Interscience Publishers Inc., New York, 1957).
- [39] J. W. Wilson, G. H. Walker and R. A. Outlaw, IEEE Trans. Electron Devices 31, 421(1984).
- [40] W. A. McKinley and H. Feshbach, Phys. Rev. 74, 1759(1948).
- [41] J. F. Janni, Atomic Data and Nuclear Data Tables 27, 147(1982).
- [42] L. Pages, E. Bertel, H. Joffre and L. Sklavenitis, Atomic Data 4, 1(1972).
- [43] G. H. Walker and E. J. Conway, Proceedings of 13th IEEE Photovoltaic Specialists Conference, 575(1978).
- [44] G. H. Walker and E. J. Conway, J. Electrochem. Soc., 1926(1978).
- [45] C. T. Sah, R. N. Noyce and W. Shockley, Proceeding of IRE 45, 1228(1957).
- [46] J. M. Dennry, R. G. Dowling, S. R. Lackman and J. W. Oliver, Space Technology Labs, EM1021 Mr-13, (1961)
- [47] M. Wolf, Proceedings of the IEEE, 51, 674(1963)
- [48] R. Y. Loo and G. S. Kamath, Proceedings of 14th IEEE Photovoltaic Specialists Conference, 1090(1980).
- [49] R. Y. Loo, G. S. Kamath and R. C. Knechtli, NASA Final Report, NAS1-15443 (1979).
- [50] R. Y. Loo, G. S. Kamath and R. C. Knechtli, NASA Final Report, NAS1-15926 (1982).
- [51] R. Y. Loo, G. S. Kamath and R. C. Knechtli, NASA Final Report, NAS1-14727 (1979).
- [52] C. S. Yeh, S. S. Li and R. Y. Loo, J. Appl. Phys. 62, 4611(1987).
- [53] C. S. Yeh, S. S. Li and R. Y. Loo, Technical Digest of 3rd International Photovoltaic Science and Engineering Conference, 627(1987).
- [54] C. S. Yeh, S. S. Li and R. Y. Loo, Proceedings of 18th Photovoltaic Specialists Conference, 657(1985).

- [55] D. T. Rover, P. A. Basore and G. M. Thorson, Proceedings of 17th Photovoltaic Specialists Conference, 703(1984).
- [56] M. J. Chen and C. Y. Wu, Solid State Electronics 28, 751(1985).
- [57] J. G. Fossum, Solid State Electronics 19, 269(1976).
- [58] J. J. Wysocki, Solar Energy 6, 104(1962).
- [59] F. X. Kneizys, LOWTRAN6 Computer Code, Hanscom AFB.
- [60] E. Y. Wang, F. T. S. Yu and W. L. Simms, Proceedings of 10th Photovoltaic Specialists Conference, 168(1973).
- [61] H. B. Serreze, Proceedings of 13th Photovoltaic Specialists Conference, 609(1977).
- [62] H. A. Vander Plas, L. W. James, R. L. Moon and N. J. Nelson, Proceedings of 13th Photovoltaic Specialists Conference, 934(1977).
- [63] M. A. Green, Solar Cells (Englewood Cliffs, New Jersey, 1982).
- [64] P. D. Demoulin, M. S. Lundstrom and R. J. Schwarz, Proceedings of 18th Photovoltaic Specialists Conference, 321(1985).
- [65] W. E. Biles and J. J. Swain, Mathematical Programing Study 11, 189(1979).
- [66] J. L. Kuester and J. H. Mize, Optimization Technique with FORTRAN (McGraw-Hill, New York, 1973).
- [67] D. K. Schroder and P. L. Meier, IEEE Trans. Electron Devices 31, 637(1984).
- [68] N. Ogasawara, S. Ochi, N. Hayafaji, M. Kato, K. Mitsue, K. Yamaraka and T. Murotani, Technical Digest of 3rd International Photovoltaic Science and Engineering Conference, 477(1987).
- [69] Annual Progress Report: Photovoltaics, DOE/CH/10093-H3.
- [70] D. A. Waterman and F. Hayes-Roth, Pattern Directed Inference Systems (Academic Press, New York, 1978).
- [71] R. T. Huang, S. K. Glandhi and J. M. Borrego, Solar Energy Material 13, 6(1986).
- [72] S. Wang, Solid State Electronics (McGraw-Hill, New York, 1966).
- [73] E. O. Kane, J. Appl. Phys. 32, 83(1961).
- [74] Elaine Rich, Artificial Intelligence (McGraw-Hill, New York, 1983).

- [75] Webster's Ninth New Collegiate Dictionary (Merriam-Webster Inc., Springfield, MA, 1983).
- [76] M. F. Lamorte and D. H. Abott, IEEE Trans. Electron Devices 31, 831(1984).
- [77] S. M. Bedair, S. B. Phatak and J. R. Hauser, IEEE Trans. Electron Devices 31, 822(1984).
- [78] C. Amano, A. Shibukawa and M. Yamaguchi, J. Appl. Phys. 58, 2780(1985).
- [79] P. Basimaji, M. Guittard, A. Rudra, J. F. Carlim and P. Gvbart, J. Appl. Phys. 62, 2103(1987).
- [80] A. K. Saxena, J. Electron. Mater. 3, 453(1982).
- [81] A. K. Saxena and M. A. Mudares, J. Appl. Phys., 58, 2795(1985).
- [82] A. K. Saxena, J. Appl. Phys. 52, 5643(1981).
- [83] G. B. Stringfellow, J. of Appl. Phys. 50, 4178(1979).
- [84] K. Seeger, Semiconductor Physics An Introduction (Springer-Verlag, New York, 1985).
- [85] A. Fortini, D. Diguët and J. Lugand, J. Appl. Phys. 41, 3121(1970).
- [86] D. L. Rode, Phys. Rev. 2, 1012(1970).
- [87] R. E. Fern and A. Onton, J. Appl. Phys. 42, 3499(1971).
- [88] S. Adachi, J. Appl. Phys. 58, R1(1985).
- [89] A. K. Saxena, Phys. Rev., B 6, 3295(1981).
- [90] J. Bardeen and W. Shockley, Phys. Rev. 77, 407(1962).
- [91] C. A. Mead and W. G. Spitzer, Phys. Rev. Lett. 11, 358(1963).
- [92] S. S. Li, Semiconductor Physical Electronics, (to be published).
- [93] L. R. Weisberg, J. Appl. Phys. 33, 1817(1962).
- [94] E. M. Conwell and M. D. Vassell, Phys. Rev. 166, 797(1968).
- [95] J. R. Hauser, M. A. Littlejohn and W. T. Lindley, Appl. Phys. Lett. 28, 458(1976).
- [96] C. M. Wolfe, G. E. Stillman and W. T. Lindley, J. Appl. Phys. 41, 3088(1970).
- [97] W. Fawcett, A. D. Boardman and S. Swain, J. Phys. C 31, 1963(1970).


- [98] A. K. Saxena, Phys. Rev. B 25, 5428(1982).
- [99] L. Birdman, M. Lax and R. Loudon, Phys. Rev. 145, 620(1966).
- [100] D. E. Aspnes, Phys. Rev. B 34, 533(1976).

BIOGRAPHICAL SKETCH

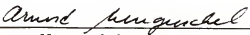
Chune-Sin Yeh was born in Taiwan in 1955. He received the B.S. degree in electrical engineering from National Cheng-Kung University, Taiwan, in 1977. He went to the College of William and Mary, Williamsburg, Virginia, in September 1981 and received an M.S. degree in applied science (computer science division) in May 1983. He also received the M.E. degree in electrical engineering from the University of Florida, in May 1985. Since then he has been working toward the Ph.D. degree in the Department of Electrical Engineering at the University of Florida.

From 1977 to 1979, he served in the Republic of China Air Force. He worked for the Department of Computer Science, Taipei City Junior College of Business, Taiwan, from 1979 to 1981. His research interests are computer modeling of multijunction solar cells and computer aided design for VLSI.

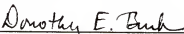
I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.


Sheng S. Li, Chairman
Professor of Electrical Engineering


I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.


Arnost Neugroschel
Professor of Electrical Engineering

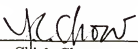
I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.


Dorothy E. Burk
Associate Professor of Electrical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.


Gijs Bosman
Associate Professor of Electrical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.


Yuan-Chieh Chow
Associate Professor of
Computer and Information Sciences

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

August 1988

Wayne H. Chen

Dean, College of Engineering

Dean, Graduate School